Hindawi
International Journal of Intelligent Systems
Volume 2023, Article ID 1666735, 21 pages
https://doi.org/10.1155/2023/1666735

Research Article

Uniformity-Comprehensive Multiobjective Optimization Evolutionary Algorithm Based on Machine Learning

Yuxuan Luan,[1] Junjiang He,[2] Jingmin Yang,[1] Xiaolong Lan,[2] and Geying Yang[3]

[1]*School of Electronics and Computer Science, University of Southampton, Southampton, UK*
[2]*School of Cyber Science and Engineering, Sichuan University, Chengdu, China*
[3]*School of Cyber Science and Engineering, Wuhan University, Wuhan, China*

Correspondence should be addressed to Junjiang He; hejunjiang@scu.edu.cn

Received 14 August 2023; Revised 26 September 2023; Accepted 26 October 2023; Published 10 November 2023

Academic Editor: Fabio Caraffini

When solving real-world optimization problems, the uniformity of Pareto fronts is an essential strategy in multiobjective optimization problems (MOPs). However, it is a common challenge for many existing multiobjective optimization algorithms due to the skewed distribution of solutions and biases towards specific objective functions. This paper proposes a uniformity-comprehensive multiobjective optimization evolutionary algorithm based on machine learning to address this limitation. Our algorithm utilizes uniform initialization and self-organizing map (SOM) to enhance population diversity and uniformity. We track the IGD value and use K-means and CNN refinement with crossover and mutation techniques during evolutionary stages. Our algorithm's uniformity and objective function balance superiority were verified through comparative analysis with 13 other algorithms, including eight traditional multiobjective optimization algorithms, three machine learning-based enhanced multiobjective optimization algorithms, and two algorithms with objective initialization improvements. Based on these comprehensive experiments, it has been proven that our algorithm outperforms other existing algorithms in these areas.

1. Introduction

Multiobjective optimization problems play a crucial role in resolving many optimization issues in the real world [1]. The essence of MOP lies in the pursuit of solutions that effectively meet the constraints imposed by multiple objective functions, as exemplified in the following equation:

$$\text{Minimise } F(\mathbf{x}) = \left(f_1(\mathbf{x}), \ldots, f_m(\mathbf{x})\right), \mathrm{x} \in \Omega, \quad (1)$$

where $\mathbf{x}$ represents the decision variables, $F(\mathbf{x})$ represents the vector of objective functions, $f_1(\mathbf{x}), \ldots, f_m(\mathbf{x})$ is each objective function, and Ω represents the decision space. Many multiobjective optimization algorithms struggle with uneven solution distribution throughout the exploration process, affecting uniformity across the search space. These issues intensify when handling high-dimensional and complex problems, potentially losing high-quality solutions and constraining the algorithm's global optimization capability. Given the complexity of these problems' objective functions, there is a pressing need for algorithms with enhanced adaptability and flexibility. Such limitations hinder their effectiveness and broader application.

In recent years, multiobjective optimization algorithms have shown significant potential in tackling complex problems and optimizing decisions in various fields, including network security [2], energy engineering [3, 4], computer vision [5], and healthcare [6]. Thus, researchers from various domains have been dedicated to exploring and solving multiobjective optimization problems over the past few decades. Currently, evolutionary algorithms offer distinctive strategies to handle complex multiobjective optimization issues with their unique global search capabilities and robust adaptability. Therefore, the universality and effectiveness of evolutionary algorithms have played an indispensable role in the study of multiobjective optimization problems, exerting a profound influence on optimization decisions.

Over the past three decades, the use of evolutionary algorithms to solve multiobjective optimization problems has received increasing attention from experts and scholars in various disciplines. It has become a powerful focus of research. This paper classifies the existing multiobjective evolutionary algorithms into four categories. The first category comprises multiobjective evolutionary algorithms based on spatial partitioning techniques, with the MOEA/D algorithm being an excellent example [7]. This algorithm transforms a complex multiobjective problem into a series of relatively more straightforward single-objective optimization problems by leveraging weight vectors to signify the objectives of each subproblem, thereby generating superior and uniformly distributed solution sets. Recent advances in research have further augmented the efficiency of this method [8–10]. The second category involves distance-based algorithms concentrating on attaining homogeneity by calculating and preserving the distance between solutions to prevent excessive aggregation or sparsity. The NSGAII algorithm exemplifies this approach [11], and recent designs of novel learning mechanisms have been refined [12–14]. The third category refers to weight-based evolutionary algorithms for multiobjective optimization. This method achieves homogeneity by integrating weights into the objective function to control the impact of each objective function on the optimization result. For example, MOEA/D-DDE ensures population homogeneity through a uniform weight vector [15]. Interestingly, recent studies such as CCGDE3 [16] and GDE3 [17] have proposed enhancements to these differential evolutionary algorithms, effectively bolstering their diversity and convergence. The final category includes evolutionary algorithms of multiobjective optimization based on stratification, such as the SPEA2 algorithm [18]. This method stratifies individuals in the population based on dominance relationships, where individuals within each stratum do not dominate each other. This technique helps in generating diverse and uniformly distributed set of solutions. Subsequent studies have further enhanced the solution capability of population SPEA2-based optimization algorithms utilizing immunity algorithms [19–21].

In the discussion of various multiobjective evolutionary algorithms (MOEAs), the significance of sample uniformity becomes apparent in influencing the algorithmic evolution process. For example, NSGAII employs crowding distance computation to ensure a uniformly distributed set of solutions. MOEA/D, on the other hand, optimizes decomposed single-objective subproblems using reference points to yield uniformly distributed solution sets. Moreover, MOEA/D-DE enhances this by incorporating differential mutation operators to ensure sample diversity during iterative computations. SPEA2 adopts a layer-based mechanism to generate diversified and uniformly distributed solution sets. While these algorithms demonstrate efficacy in achieving a uniform distribution of solutions, it has been observed that they often focus on uniformity during a specific phase, overlooking the significance of maintaining this uniform distribution throughout the evolutionary process. This oversight can lead to potential biases or inconsistencies in the solution distribution. Therefore, rather than concentrating on uniformity in just one phase, it is more beneficial to continuously monitor and adjust the uniformity strategy throughout the entire evolutionary process. The primary objective of this research is to explore methods for maintaining solution uniformity throughout the entire algorithm and to consider dynamically adjusting strategies throughout the evolution to ensure this aim is achieved. The research aims to maintain solution uniformity by dynamically adjusting strategies throughout the evolutionary process.

In light of our profound understanding of the importance of uniformity throughout the evolutionary process, we introduce a novel multiobjective evolutionary algorithm: uniformity-comprehensive multiobjective optimization evolutionary algorithm based on machine learning (MOEA-UCML). This algorithm emphasizes uniformity during the population initialization phase and maintains a consistent solution distribution throughout the evolutionary journey. We incorporate a uniformity mechanism right from the population initialization to ensure a comprehensive search foundation from the outset. To sustain solution uniformity throughout the evolutionary process, we adopt a dynamic convergence determination mechanism and delineate the process into three distinct phases, guiding the balance between search efficiency and solution quality. Furthermore, to address the complexity and dynamism of the problems, we leverage machine learning techniques to dynamically adjust the mutation strategy throughout the process, enhancing the algorithm's optimization prowess. The main contributions of our proposed MOEA-UCML are as follows:

(1) A unique multistage optimization strategy is proposed, diverging from conventional methods that achieve uniform distribution at specific algorithm stages. The strategy integrates a uniformity mechanism during population initialization and consistently adopts SOM techniques throughout the optimization process. This continuous application of SOM ensures that the population upholds a uniform distribution of solutions, guaranteeing solution diversity.

(2) A dynamic convergence detection mechanism is incorporated to guide the algorithm's progress. This method diverges from traditional approaches that use fixed-stopping criteria. Instead, it utilizes a distinct heterogeneity calculation method paired with the IGD index for evaluating population diversity and algorithm convergence. The algorithm ensures maintained solution diversity in a dynamically changing optimization environment, bolstering the algorithm's robustness and flexibility.

(3) Integrating a convolutional neural network (CNN) into the MOEA-UCML algorithm distinguishes it from conventional multiobjective optimization algorithms by utilizing a CNN, guiding mutation operations to achieve a uniform distribution on the Pareto frontier. This method enhances the effectiveness of discovering high-quality and uniformly distributed solutions in high-dimensional search spaces, significantly improving the algorithm's optimization performance.

(4) The MOEA-UCML algorithm is compared with 13 other algorithms, including traditional multiobjective optimization algorithms, machine learning-based improved multiobjective optimization algorithms, and population initialization-based improved multiobjective optimization algorithms. Experiments covering 25 widely used test problems from ZDT, DTLZ, WFG, and UF verify the algorithm's significant superiority over the existing solutions' distribution. Experimental analysis of the algorithm's uniform initialization strategy and neuro-clustered optimization strategy further validates the robustness and universality of this approach.

The subsequent sections of this paper are structured as follows. Section 2 reviews the relevant literature and provides a foundational understanding of multiobjective optimization algorithms. Section 3 illustrates in detail the proposed methodology, MOEA-UCML. Section 4 furnishes experimental outcomes, supplemented by a comprehensive analysis. Lastly, Section 5 encapsulates the salient findings of this study and suggests potential paths for future research.

2. Related Work

While traditional optimization techniques might need to be revised when addressing intricate challenges, the integration of machine learning offers a paradigm shift in optimization research. This section delves into current research on strategies for population uniformity, optimization of mutation strategies guided by machine learning, and the formulation and improvement of evaluation metrics.

2.1. Population Uniformity. For the uniformity of populations in multiobjective optimization problems, researchers have delved into and enhanced population uniformization strategies from various perspectives. These perspectives encompass population diversity, solution space coverage, exploration breadth, exploitation depth, and adaptability. In 2016, Elsayed et al. [22] designed a series of predefined sequences to customize the initial population, covering the entire solution space, thus enhancing the diversity of the population and the potential of the solutions. In 2019, Deniz and Kiziloz [23] proposed an initial population generation method based on information gain ranking (IGR) to improve the initial performance of evolutionary algorithms in feature subset selection (FSS). This method significantly optimized the execution time and learning performance of algorithms on medium and large datasets and was proven to be an efficient strategy for generating initial populations. In 2022, Huang and Zhang [24] explored the uniformity issue in multiobjective optimization and proposed a strategy based on adaptive competitive swarm optimization. This method introduces an enhanced competition mechanism and learning scheme and incorporates external archiving and its maintenance strategy to ensure solution diversity and convergence. Compared to other leading methods, this strategy demonstrates significant superiority in addressing multiobjective optimization problems, especially ensuring a uniform distribution of solutions. Similarly, in 2022, Wang and associates [25] introduced an evolutionary algorithm based on particle swarm prediction and predictive adjustment strategies. This method merges information from the prechange population with predictive adjustments to the new environment, aiming to maintain high-quality and uniform solutions in dynamic environments. Against other approaches, this strategy shows remarkable advantages in uniformity and environmental responsiveness. Summarizing the above research, maintaining and enhancing population uniformity in multiobjective optimization algorithms are pivotal.

2.2. Machine Learning-Guided Variation. In research on multiobjective optimization guided by machine learning, researchers have fully leveraged the advantages of machine learning, improving the efficiency and stability of solutions to multiobjective optimization problems and providing valuable theoretical tools and practical references for the field. In 2019, Lv and colleagues [26] devised a particle swarm strategy for multiobjective optimization, incorporating proactive learning to amplify the model's approximation capability. They also introduced a hybrid mutation sampling method based on simulated evolution to guarantee solution diversity. This strategy boosts the algorithm's operational efficiency and reduces the complexity of finding solutions. In 2021, Zou et al. [27] proposed a reinforcement learning approach named RL-DMOEA. It estimates the severity of environmental changes through corresponding changes in the target space, thereby relocating individuals. This strategy proved effective in real-world problems, further demonstrating its superiority in handling dynamic multiobjective optimization problems. That same year, Li's team [28] integrated a methodology founded on modular neural networks to predict population dynamics effectively while factoring in evolutionary strategies' multiobjective optimization attributes. In scenarios characterized by dynamic optimization changes, this approach optimizes the algorithm's adaptability and prediction accuracy, standing out as a proficient strategy for dynamic multiobjective optimization. In 2022, Liu and his team [29] incorporated a dual-layer sampling strategy to counteract the search space's exponential expansion. They unveiled an offspring sampling method to generate promising candidate solutions. When dealing with large-scale decision variables, the algorithm optimizes search efficiency and solution quality, marking its place as an effective multiobjective optimization strategy. Also, in 2022, Tian et al. [30] proposed a method using deep reinforcement learning (DRL) that adaptively selects operators through deep neural networks, effectively solving the operator selection problem in evolutionary multiobjective optimization and enhancing the efficiency of solving such problems.

2.3. Evaluation Indicators. In the performance evaluation and algorithm efficiency optimization for multiobjective optimization problems, numerous researchers have

proposed improvements to evaluation metrics such as inverted generational distance (IGD) and related solution generation strategies. In 2020, Cai et al. [31] proposed an improvement method based on a grid-based inverse generational distance (G-IGD), optimizing the calculation of IGD by dividing the reference set into multiple grids. G-IGD was used as a performance indicator to evaluate the performance of multiobjective optimization and multiobjective optimization algorithms, making performance evaluation and comparison of different algorithms more effective. In addition, in 2020, Chen et al. [32] proposed a generalized hypervolume contribution (GHC) method. GHC provides greater flexibility in various problem scenarios by introducing user-defined weight vectors. GHC effectively guides the initialization of the population in multiobjective optimization problems, thus improving the quality of the solution and algorithmic efficiency and reducing the complexity of solutions. In 2022, Han and his team [33] focused on three core metrics in decision space: the inverse generational distance (IGD), the multimodal inverse generational distance (IGDM), and the hypervolume (HV). These meticulous evaluation tools quantify algorithm performance, convergence, and diversity within the decision space and offer insightful perspectives and methodological foundations for assessing and comparing multimodal optimization algorithms. In 2023, Yan et al. [17] introduced an improved inverse intergenerational distance (MIGD) metric. MIGD considers the importance of each solution in the reference set when calculating the distance between the solution set and the reference set. This new measurement method exhibits better performance and stability when dealing with dynamic multiobjective optimization problems. That same year, Song and colleagues [34] adopted the IGD + metric, reflecting both algorithms' convergence and diversity performance. Compared to the original IGD metric, the performance comparisons derived from IGD + consistently align with the Pareto dominance relationship.

3. Proposed Approach

In response to the requirements for population uniformity and diversity in multiobjective optimization problems, this paper presents a novel multiobjective optimization evolutionary algorithm based on uniformity neural network (MOEA-UCML). Figure 1 illustrates the working framework of MOEA-UCML. The algorithm uses uniform initialization and SOM strategy to enhance population diversity and uniformity and then employs IGD-driven transition to assess these qualities. If standards are not satisfied, it amplifies diversity using population diversity enhancement, and once balanced, it refines solution quality using neuro-clustered optimization.

3.1. Uniform Initialization Strategy. MOEA-UCML incorporates a uniform initialization strategy. Initially, we created an initial population using a random generation method. This step aims to cover the solution space as much as possible to enhance the diversity of solutions. Subsequently, we adopt a nondominated sorting strategy to evaluate and rank the population. In traditional nondominated sorting algorithms, the specific expression for the spacing among solutions of the same rank is provided in the following equation:

$$P_{t+1} = \mathrm{Min}\left(P_t + R_t\right), \tag{2}$$

where P_{t+1} denotes the generation population $(t+1)$, P_t represents the generation population t, R_t means the population of offspring generated in generation t, and Min indicates the individuals chosen for generation $(t+1)$, prioritized according to their classification.

This study introduces a novel initial population generation strategy based on uniform selection from each Pareto front, as shown in Figure 2. The new strategy departs from merely selecting individuals with top ranking in the population, instead opting for a uniform selection of individuals across each front, which is precisely detailed in the following equation:

$$P_{t+1} = \mathrm{Avg}\left(P_t + R_t\right), \tag{3}$$

where Avg refers to the selection of two uniformly distributed individuals from each Pareto front after the population classification for inclusion into the next generation in the initial stage. Concurrently, the crowding distance calculation method is used to ensure the uniformity of the population, which is expressed in the following equation:

$$L[i]_d = L[i]_d + \frac{\left(L[i-1]_m - L[i-1]_m\right)}{\left(f_m^{\max} - f_m^{\min}\right)}, \tag{4}$$

where $L[i]_d$ denotes the crowding distance of any individual, $L[i+1]_m$ is the value of the j-th objective function of the i-th individual, and $f_m^{\max}$ and $f_m^{\min}$ represent the maximum and minimum values of the j-th objective function in the set, respectively.

Finally, the resulting initial population served as the input for subsequent self-organizing map (SOM) optimization strategies, expecting to enhance population uniformity and diversity further. To elaborate on the specific implementation process of the uniform initialization strategy proposed in this paper, we provide a detailed step-by-step description of Algorithm 1.

In essence, the uniform initialization strategy not only constructs the diversity and uniformity of the solution space but also ensures the quality of the solutions. By this means, we successfully realize the uniform distribution of solutions within each nondominated layer, thereby guaranteeing the global distribution and diversity of the solutions.

3.2. SOM Optimization Strategy. Following the uniform initialization strategy of MOEA-UCML, the algorithm's next step focuses on ensuring population uniformity and diversity throughout the search process. To address this, MOEA-UCML integrates an optimization strategy based on self-organizing map (SOM) while uniformly selecting individuals from the Pareto front. Specifically, we identify the closest neuron within the SOM for each input sample, termed

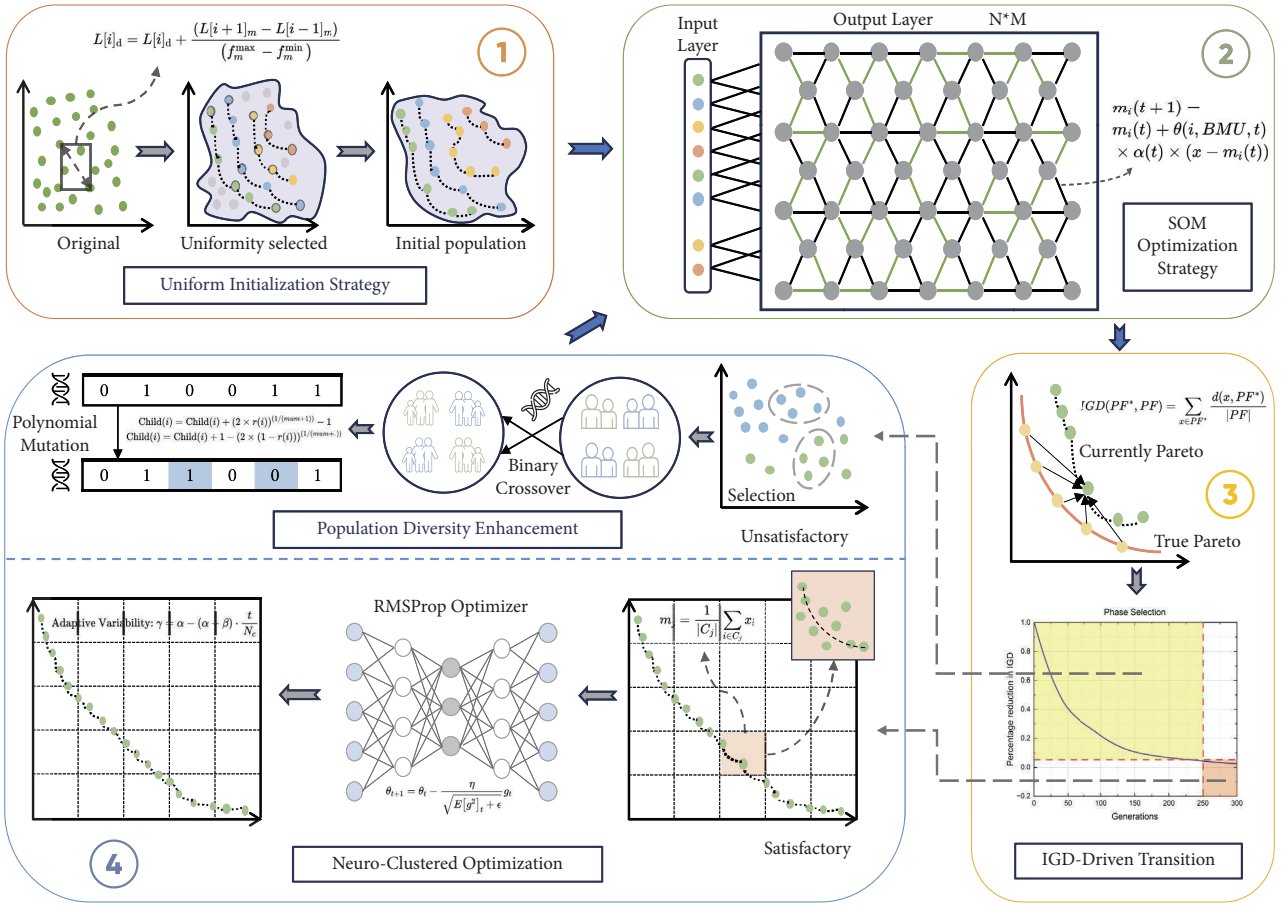

Figure 1: Framework of MOEA-UCML.

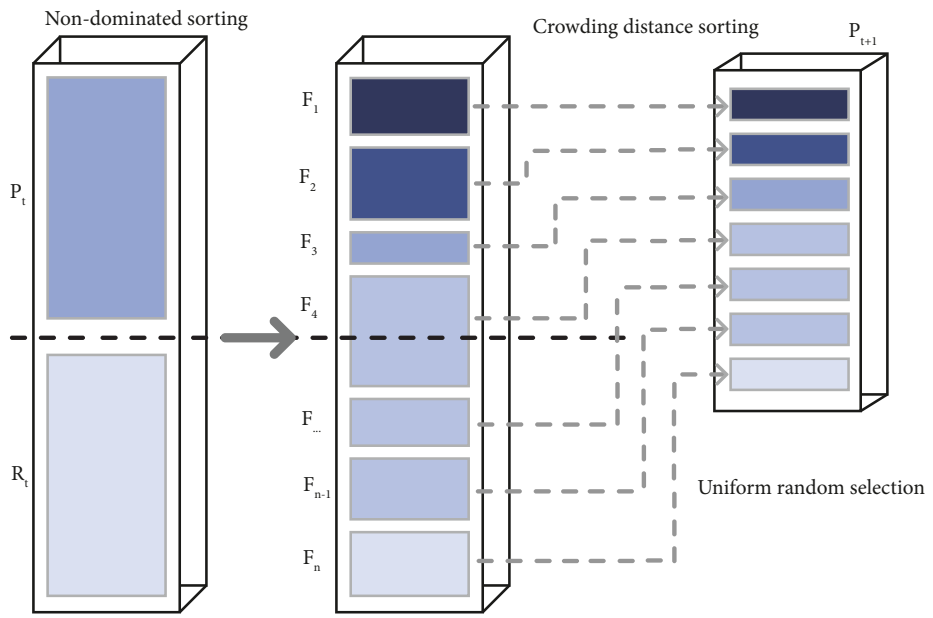

Figure 2: Nondominated sorting improvements.

Input: N (population size), V (number of decision variables), M (number of targets), gen_1 (number of first stage iterations)
Output: $P^{(\text{gen}_1)}$ (phase 1 population).
(1) **P** ⟵ Random (**N**, **V**)
(2) **P** ⟵ Non Domination Sort (**P**, **V**, **M**)
(3) [**Init Unevenness**, ~] ⟵ Train and Measure SOM (**P**)/* Record initial distance */
(4) for g ⟵ 2 to $\text{gen}_1 - 1$ do
(5) **P′** ⟵ Random (**N**, **V**)
(6) **P** ⟵ Non Domination Sort (**P** ∪ **P′**, **V**, **M**)
(7) **P** ⟵ Uniform extraction of individuals from each layer
(8) end for
(9) return **P**

Algorithm 1: Initialize variables.

the best matching unit (BMU). The BMU pinpoints a probable solution location and offers vital guidance for subsequent optimization, as illustrated in the following equation:

$$\text{BMU}(x) = \arg\min_i \left\| x - m_i \right\|, \tag{5}$$

where x denotes the input sample and $\min_i \|x - m_i\|$ refers to the neuron with the smallest Euclidean distance. Subsequently, we employ the SOM's unsupervised neural network properties to adjust the neurons' weight vectors. This step enables us to optimize the location of solutions, further enhancing the diversity of solutions. The update rule is represented in the following equation:

$$m_i(t+1) = m_i(t) + \theta(i, \text{BMU}, t) \times \alpha(t) \times (x - m_i(t)), \tag{6}$$

where t represents the current iteration step and $\alpha(t)$ is the learning rate, which gradually decreases with time. $\theta(i, \text{BMU}, t)$ is the neighborhood function that determines how the weight vector of neuron i should be updated based on BMU. As the BMU weights are updated, the weights of neighboring neurons are also updated. In particular, the closer a neuron is to the BMU, the more significant its weight adjustments will be. This step is designed to ensure the uniformity of the solutions by guaranteeing that each solution has enough space to explore within the solution space. According to equation (6), the update process is governed by a control known as the neighborhood function, described in detail in the following equation:

$$\theta(i, \text{BMU}, t) = \exp\left(-\frac{\left\| r_i - r_{\text{BMU}} \right\|^2}{2\sigma(t)^2} \right), \tag{7}$$

where r_i and r_{BMU} represent the lattice positions of neuron i and BMU in the SOM, while $\|.\|^2$ denotes the Euclidean distance. The parameter $\sigma(t)$ is a positive value that varies over time and determines the width of the neighborhood, typically decreasing gradually with time. This mechanism effectively merges global and fine-grained search, making the optimization process more comprehensive and efficient. Finally, we continuously optimize and update the population through nondominated sorting and a feedback mechanism to improve search efficiency. To explain the SOM optimization strategy proposed in this study, we describe the specific implementation process in detail in Algorithm 2.

3.3. IGD-Driven Transition. In MOEA-UCML, the IGD-driven transition is a linchpin that provides a precise trajectory for the optimization journey. The IGD denoted by equation (8) functions as a metric to measure the effectiveness of multiobjective optimization algorithms.

$$\text{IGD}(PF^*, PF) = \sum_{x \in PF^*} \frac{d(x, PF^*)}{|PF|}, \tag{8}$$

where PF signifies a point set distributed uniformly on the true Pareto front of the test function and P^* encompasses the optimal front set derived from the algorithm. $d(x, \text{PF}^*)$ denotes the nearest Euclidean distance between individuals in PF and P^*, while |PF| quantifies the count of uniformly distributed points in PF. This equation measures the mean distance between algorithmically derived solutions and their true counterparts, furnishing a composite evaluation of population quality and diversity.

When the population falls short of the predefined IGD benchmark, MOEA-UCML focuses on the population diversity enhancement phase, underscoring the importance of population diversity. The freshly generated offspring undergo a uniformization process through SOM, safeguarding their even spread across the entire solution realm. Following uniformization, the population undergoes a subsequent assessment via the IGD evaluation framework, confirming that our optimization trajectory is closely aligned with the genuine Pareto front.

Upon achieving the stipulated IGD criteria, the algorithm delves into the neuro-clustered optimization phase. This phase harnesses neural network clustering techniques to meticulously refine and optimize the population, ensuring its prime distribution in the objective function space.

3.4. Mutation Strategy

3.4.1. Population Diversity Enhancement. During the population diversity enhancement phase of the MOEA-UCML algorithm, simulated binary crossover and polynomial mutation stand out as pivotal operations designed to bolster population diversity and uniformity further. The simulated binary crossover emulates crossover operations in a binary encoding system yet finds its application in real-number encoding. When generating offspring, this operation takes

Input: P (population)
Output: $P^{(\text{Weights})}$ (offspring output), Unevenness (Euclidean distance).
(1) **NET** ⟵ Initialize a new SOM network
(2) **NET** ⟵ train (**NET**, **P**)/∗ Training the SOM network ∗/
(3) **P** ⟵ **NET**.IW {pdist (1)}
(4) **Unevenness** ⟵ sum (pdist (**P**))/∗ Calculation of inhomogeneity ∗/
(5) return [**Unevenness**, **P**]

Algorithm 2: Train and measure SOM.

into account the difference between the two parents, ensuring that the newly spawned offspring enjoy a broader distribution in the solution space. On the other hand, polynomial mutation serves as a fine-tuning operation governed by equations (9) and (10).

$$\text{Child}\ (i) = \text{Child}\ (i) + (2 \times r(i))^{(1/\mu_m+1)} - 1, \tag{9}$$

$$\text{Child}\ (i) = \text{Child}\ (i) + 1 - (2 \times (1 - r(i)))^{(1/\mu m+1)}, \tag{10}$$

where Child (i) represents the i-th element of the offspring, and the choice of which mutation formula to apply depends on the value of $r(i)$. μ_m acts as a parameter in the polynomial to regulate the intensity of the mutation. This mutation approach, employing a predefined polynomial distribution, facilitates precise fine-tuning of the selected gene, ensuring that the population executes a meticulous exploration in the solution space. By integrating these two strategies, the population diversity enhancement phase ensures that the population maintains diversity and conducts a comprehensive search throughout the solution space.

3.4.2. Neuro-Clustered Optimization. In the IGD-driven transition phase of the algorithm, we evaluate the uniformity and diversity of the population. Once the IGD value meets the predetermined criteria, we transition to the neuro-clustered optimization phase, aiming to refine the nonuniform regions within the population further.

Initially, we employ a K-means-based approach to pinpoint the most nonuniform areas in the population. We can express this through a formula by randomly selecting K data points as the initial cluster centers and subsequently classifying each data point based on its distance to these centers by the following equation:

$$C(i) = \arg\min_j \left\| x_i - m_j \right\|, \tag{11}$$

where x_i denotes the data point, m_j represents the center of the j-th cluster, and $C(i)$ signifies the cluster to which data point i belongs. The notation $\|.\|$ indicates the Euclidean distance. We then update each cluster's center to equal the mean value of all data points within that cluster, as expressed in the following equation:

$$m_j = \frac{1}{\left| C_j \right|} \sum_{i \in C_j} x_i, \tag{12}$$

where $C(j)$ is the j-th cluster, with $|.|$ indicating the size of the set, namely, the number of data points in the cluster. We iteratively apply equations (11) and (12) until the cluster centers stabilize. The K-means algorithm allows us to pinpoint the most nonuniform regions in the population, maintaining the advantages of the population while enhancing the comprehensiveness of the population frontier, offering decision makers an expanded set of solution choices.

Having identified these nonuniform clusters, we incorporate a neural network model to forecast the mutation outcomes of the population's individuals. During the neural network's parameter update process, we employ the RMSProp optimizer, represented by the following equation:

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{E\left[g^2\right]_t + \epsilon}} g_t, \tag{13}$$

where θ denotes the neural network parameters, η is the learning rate, g_t signifies the gradient at step t, $E[g^2]_t$ stands for the decaying average of past gradient squares, and ϵ is a tiny constant to prevent division by zero. We then apply the trained neural network to the nonuniform clusters previously identified using K-means, fine-tuning these individuals predictively. To augment the adaptiveness of the search, we also adopt an adaptive mutation rate strategy, depicted by the following equation:

$$\gamma = \alpha - (\alpha - \beta) \cdot \frac{t}{N_c}, \tag{14}$$

where α sets the initial mutation rate, β determines the final mutation rate, t is the current iteration count, and N_c is the number of the most nonuniform clusters. This design means that as the iteration progresses, we gradually reduce the mutation rate, making the algorithm lean more towards local search. In the end, we reintegrate the mutated individuals into the original population, replacing the least uniform clusters with these newly generated individuals. We describe the detailed steps in Algorithm 3.

Through the neuro-clustered optimization strategy, we merged the strengths of deep learning and genetic algorithms, effectively addressed the population's nonuniformity, and simultaneously enhanced the quality of population solutions. This offers decision makers a more comprehensive and high-quality array of choices.

3.5. The Complete MOEA-UCML. To illustrate the process of implementation of the algorithm MOEA-UCML more intuitively, the pseudocode of the algorithm is given in Algorithm 4. The algorithm is explained in detail as follows.

Input: P (population), V (number of decision variables), M (number of targets), gen_3 (number of third stage iterations), k (cluster)
Output: $P^{(\text{gen}_3)}$ (phase 3 population).
(1) **P** ⟵ Crossover work/∗ Training neural network ∗/
(2) **NET** ⟵ train (**P**, **P**, **Layers**, **Options**)
(3) [**C**, **Num**] ⟵ K-means (**P**, k)/∗ K-means finds the most inhomogeneous cluster ∗/
(4) for g ⟵ 1 to size (**C**) − 1 do
(5) **C** ⟵ predict (**NET**, **C**);
(6) **C′** ⟵ **C*****Mutation**
(7) **C** ⟵ max (0, min (1, **C′**))
(8) **Mutation** ⟵ Calculate the Mutation by using equation (11)
(9) end for
(10) **P** ⟵ **P** (**P** (**Num**)! = **C** (**Num**)) ∪ **C**/∗ Replace the variant part ∗/
(11) return **P**

Algorithm 3: Mutation NN.

3.5.1. Lines 1–3. Initially, parameters are set according to the objective problem and population variables are initialized. Simultaneously, the population is ranked based on the nondominance principle.

3.5.2. Lines 4–11. From the initial generation phase to the fitness phase, this stage constitutes the main loop of the algorithm. First, tournament selection selects the most suitable parents for reproduction based on each individual's fitness. Then, during the genetic operation stage, these parents are altered using crossover and mutation to produce offspring. Next, these offspring are once again ranked according to the nondominance principle, and chromosomes for replacement are selected based on their priority at the boundary. Finally, the IGD value of the current boundary is calculated, representing the population's distance from the ideal solution. If this value exceeds the preset threshold, this phase is prematurely terminated and transitions to the next stage.

3.5.3. Lines 12–16. In the algorithm optimization phase, the algorithm enters the second main loop, running from the fitness phase to the uniformity fine-tuning phase. During this stage, the individual reproduction method no longer employs traditional genetic operations for reproduction but adopts a neural network mutation. This method combines the advantages of deep learning and genetic algorithms, enabling the algorithm to find potential quality solutions more precisely, thus better guiding the algorithm's search direction and allowing the algorithm to find superior solutions more rapidly during the search process.

3.5.4. Line 17. Once the maximum number of generations is reached, the algorithm returns the final population, representing the set of solutions found by the algorithm.

4. Experimental Results

This section evaluates the MOEA-UCML algorithm's performance through comprehensive experiments. We detail the test functions, performance evaluation metrics, and experimental parameter settings. Additionally, we compare the effectiveness of the MOEA-UCML algorithm with the state-of-the-art multiobjective evolutionary algorithms and analyze the algorithm's sensitivity to parameters.

4.1. Test Problem Functions. To comprehensively evaluate the performance of the proposed algorithm, we selected 25 unconstrained test functions commonly used for multiobjective optimization from the ZDT ($n = 30$ or 10), DTLZ ($n = 7$ or 12), WFG ($n = 12$), and UF ($n = 30$) problem sets. These test functions encapsulate various complex characteristics such as convexity, nonuniformity, discontinuity, unimodality, multimodality, and local optima, allowing us to test the algorithm's efficacy across multiple dimensions. Specifically, the ZDT series targets two objective dimensions. In comparison, the DTLZ and WFG series focus on three objective dimensions, and the UF series evaluates the algorithm's performance on multiple objective dimensions. By employing these widely accepted test function series, we furnished an extensive benchmark for assessing our proposed algorithm in multiobjective optimization problems. We set the population size for all these algorithms at 200, with a maximum function evaluation count of 10,000. The class value for the K-means algorithm is set to 10, indicating that we divide the population frontier into ten distinct regions. Additionally, we set the parameter "um" at 20, designed the CNN network with three fully connected layers, fixed the learning rate at 0.01, capped the training at 200 epochs, trained with a batch size of 64 samples, and designated α and β at 0.05 and 0.01, respectively. We sourced the other parameter settings for each algorithm from their respective reference literature. We implemented all multiobjective optimization algorithms on the PlatEMO platform. To reduce the variability due to randomness, we executed each test function 30 times. We then computed the mean and standard deviation (std) of the IGD values from these runs for subsequent comparison.

4.2. Performance Evaluation Metrics and Parameter Settings. In this study, we employ the IGD metric to comprehensively evaluate the comparative algorithm's capabilities, assessing its efficacy in optimizing the objectives within a specific

Input: N (population size), Q (targeted questions)
Output: $P^{(g_{\max})}$ (final population).
(1) (**V**, **M**) ⟵ Parameter Settings (**Q**)
(2) **P** ⟵ Initialize Variables (**N**, **V**, **M**, $\mathbf{gen_1}$)
(3) **P** ⟵ Non Domination Sort (**P**, **V**, **M**)
(4) for g ⟵ $\mathbf{gen_1}$ to $\mathbf{gen_2}$ do
(5) [∼, **P**] ⟵ Train and Measure SOM (**P**)
(6) **Parent** ⟵ Tournament Selection (**P**)/∗ Select suitable sires for breeding ∗/
(7) **P′** ⟵ Genetic Operator (**Parent**, **V**, **M**)/∗ Cross mutation to produce offspring ∗/
(8) **P** ⟵ Non Domination Sort (**P′**, **V**, **M**)
(9) **P** ⟵ Replace Chromosome (**P**, **V**, **M**)/∗ Select individuals with frontier priorities ∗/
(10) **IGD** ⟵ Calculating the IGD value of the current frontier
(11) if **IGD** > **Limited Values** then
(12) break;
end if
end for
(13) for g ⟵ $\mathbf{gen_2}$ to $\mathbf{g_{max}}$ do
(14) [∼, **P**] ⟵ Train And Measure SOM (**P**)
(15) **Parent** ⟵ Tournament Selection (**P**)
(16) **P** ∗ ⟵ Mutation NN (**Parent**, **V**, **M**)/∗ neuro-clustered optimization strategy ∗/
(17) **P** ⟵ Non Domination Sort (**P** ∗ , **V**, **M**)
(18) **P** ⟵ Replace Chromosome (**P**, **V**, **M**)/∗ Select individuals with frontier priorities ∗/
end for
(19) return **P**

Algorithm 4: Framework of MOEA-UCML.

problem domain—the definition of IGD in equation (8). The lower the IGD value, the higher the quality of the approximate solution obtained by the multiobjective optimization algorithm in this experiment. Additionally, we incorporate the hypervolume (HV) metric to complement our evaluation process. The hypervolume metric captures the volume of the objective space dominated by a Pareto front approximation, representing a quantifiable coverage measure. The mathematical representation of HV is given by

$$\mathrm{HV}(A, r) = \mathrm{volume}\left(\bigcup_{a \in A} [a, r]\right), \tag{15}$$

where A is an approximate set of Pareto fronts, r is a reference point, usually chosen to be worse than all the target values in A, and $[a, r]$ denotes the hypercube determined by the solution a and the reference point r. The larger the value of HV is, the better the algorithm performs. In other words, a larger HV means that the approximate solution of the Pareto front occupies a larger target space and is closer to the true Pareto front.

In this experiment, the clustering of K-means of MOEA-UCML is adopted, with k set at 10, and the threshold to enter the later phase of the fine-tuning of neural network prediction will be set differently according to different problems. The algorithm comparison includes eight traditional multiobjective optimization algorithms: CCGDE3 [16], GDE3 [17], IBEA [35], HypE [36], MOEA/D [7], MOEA/D-DE [15], NSGAII [11], and NSGAIII [37], as well as three improved multiobjective optimization algorithms based on advanced machine learning: MOEA/D-DQN and its two variants MOEA/D-OP1 and MOEA/D-OP2 [30]. Additionally, it includes two multiobjective optimization algorithms improved by advanced goal initialization: CMOPSO [38] and MOEA/D-AAWN [39].

4.3. Performance Comparison of MOEA-UCML with Various Multiobjective Optimization Algorithms

4.3.1. Performance Comparison with Eight Traditional Multiobjective Optimization Algorithms. In the field of multiobjective optimization, to validate the performance of our proposed MOEA-UCML algorithm, we initially compared its performance with six classic multiobjective optimization algorithms (specifically, CCGDE3, GDE3, IBEA, MOEA/D, MOEA/D-DE, and NSGAIII) using ZDT and DTLZ test functions. As shown in Tables 1 and 2, we present the performance results of all algorithms for each test function, highlighting the optimal results in bold. The final row of the table represents the statistical comparison of the performance between the MOEA-UCML algorithm and the other algorithms using the two-sided Wilcoxon rank-sum test. The symbols "–," "+," and " ≈ " denote cases where the mean performance of the MOEA-UCML algorithm is less than, more significant, or approximately equal to that of the other algorithms, respectively.

From Table 1, it can be inferred that among all the ZDT and DTLZ test functions, the MOEA-UCML algorithm outperforms CCGDE3, GDE3, IBEA, MOEA/D, MOEA/D-DE, and NSGAIII on multiple test functions. Regarding the IGD results, MOEA-UCML predominantly outperforms CCGDE3, GDE3, IBEA, MOEA/D, MOEA/D-DE, and NSGAIII in 12 test problems. It is inferior to NSGAIII on ZDT3 and DTLZ2 and less efficient than MOEA/D-DE on ZDT6. In particular,

Table 1: The IGD comparison results of MOEA-UCML with six traditional multiobjective optimization algorithms on the ZDT and DTLZ problem sets.

Problem	CCGDE3	GDE3	IBEA	MOEA/D	MOEA/D-DE	NSGAIII	MOEA-UCML
ZDT1	$9.6880e-1$ $(1.17e-1)-$	$1.2459e+0$ $(1.11e-1)-$	$2.9742e-2$ $(7.25e-3)-$	$1.5069e-1$ $(8.21e-2)-$	$8.5603e-1$ $(1.53e-1)-$	$8.2198e-2$ $(9.43e-3)-$	**$1.8474e-2$ $(3.55e-3)$**
ZDT2	$1.5499e+0$ $(2.32e-1)-$	$1.8394e+0$ $(2.22e-1)-$	$1.4673e-1$ $(1.21e-1)-$	$5.4373e-1$ $(7.82e-2)-$	$7.2501e-1$ $(2.00e-1)-$	**$4.4203e-2$ $(4.49e-2)\approx$**	$4.8668e-2$ $(7.26e-2)$
ZDT3	$6.0273e-1$ $(9.13e-2)-$	$9.5841e-1$ $(1.21e-1)-$	$1.8987e-2$ $(7.33e-3)-$	$1.6875e-1$ $(5.59e-2)\approx$	$4.6445e-1$ $(9.16e-2)-$	$1.9992e-2$ $(9.15e-3)-$	**$1.6742e-2$ $(5.00e-3)$**
ZDT4	$1.8313e+1$ $(6.49e+0)-$	$3.1170e+1$ $(4.68e+0)-$	$5.1140e-1$ $(1.95e-1)-$	$4.7335e-1$ $(2.10e-1)-$	$3.4854e+0$ $(1.26e+0)-$	$4.9648e-1$ $(2.55e-1)-$	**$6.0438e-2$ $(8.63e-2)$**
ZDT6	$4.8964e+0$ $(7.29e-1)-$	$1.2947e-1$ $(2.21e-1)-$	$4.3921e-2$ $(2.08e-2)-$	$7.8566e-2$ $(3.26e-2)-$	**$2.9365e-2$ $(6.70e-2)+$**	$1.7601e-1$ $(6.01e-2)-$	$6.7818e-2$ $(4.50e-2)$
DTLZ1	$7.8665e+0$ $(5.33e+0)-$	$4.5020e+0$ $(2.11e+0)-$	$2.3030e-1$ $(9.94e-2)-$	$2.3515e-1$ $(2.29e-1)-$	$7.0980e-1$ $(1.00e+0)-$	$2.2895e-1$ $(2.31e-1)\approx$	**$2.2292e-1$ $(2.89e-1)$**
DTLZ2	$1.2348e-1$ $(1.69e-2)-$	$8.2992e-2$ $(3.64e-3)-$	$8.0445e-2$ $(2.65e-3)-$	$5.4924e-2$ $(1.97e-4)+$	$7.7776e-2$ $(1.36e-3)-$	**$5.4893e-2$ $(1.78e-4)+$**	**$6.8675e-2$ $(2.47e-3)$**
DTLZ3	$8.8162e+1$ $(2.82e+1)-$	$3.4853e+1$ $(2.82e+1)-$	$6.5694e+0$ $(3.65e+0)-$	$1.7614e+1$ $(9.43e+0)-$	$1.5078e+1$ $(1.96e+1)-$	$9.0228e+0$ $(4.13e+0)-$	**$6.2585e+0$ $(3.02e+0)$**
DTLZ4	$2.8018e-1$ $(1.51e-1)-$	$1.0676e-1$ $(2.11e-2)\approx$	$1.0905e-1$ $(1.58e-1)\approx$	$4.5594e-1$ $(3.67e-1)-$	$2.1313e-1$ $(6.39e-2)-$	$2.7941e-1$ $(2.71e-1)-$	**$1.0003e-1$ $(1.95e-1)$**
DTLZ5	$5.9445e-2$ $(2.59e-2)-$	$8.9611e-3$ $(1.00e-3)-$	$1.6131e-2$ $(1.40e-3)-$	$3.2233e-2$ $(9.20e-4)-$	$1.4328e-2$ $(4.44e-4)-$	$1.2282e-2$ $(1.27e-3)-$	**$5.8972e-3$ $(2.45e-4)$**
DTLZ6	$2.2053e+0$ $(7.67e-1)-$	$1.5889e+0$ $(9.01e-1)-$	$2.6482e-2$ $(4.06e-3)-$	$1.1828e-1$ $(2.59e-1)-$	$1.5634e-2$ $(8.71e-3)-$	$1.8705e-2$ $(2.93e-3)-$	$5.8490e-3$ $(2.97e-4)$
DTLZ7	$2.1361e+0$ $(6.64e-1)-$	$2.8692e+0$ $(8.13e-1)-$	**$8.1301e-2$ $(5.45e-2)+$**	$1.7457e-1$ $(1.19e-1)-$	$4.7254e-1$ $(1.13e-1)-$	$1.0168e-1$ $(5.56e-2)\approx$	$1.0100e-1$ $(5.32e-2)$
$-/+/\approx$	12/0/0	11/0/1	10/1/1	10/1/1	11/1/0	8/1/3	

Table 2: The HV comparison results of MOEA-UCML with six traditional multiobjective optimization algorithms on the ZDT and DTLZ problem sets.

Problem	CCGDE3	GDE3	IBEA	MOEA/D	MOEA/D-DE	NSGAIII	MOEA-UCML
ZDT1	$3.3698e-2$ $(3.57e-2)-$	$2.4938e-2$ $(3.69e-2)-$	$7.0034e-1$ $(4.26e-3)\approx$	$5.2826e-1$ $(7.49e-2)-$	$2.5064e-1$ $(9.52e-2)-$	$6.9711e-1$ $(4.31e-3)-$	**$7.1421e-1$ $(1.33e-3)$**
ZDT2	$3.0475e-1$ $(7.53e-2)-$	$2.3959e-1$ $(2.14e-2)-$	$3.1323e-1$ $(9.03e-2)-$	$1.0906e-1$ $(2.76e-2)-$	$4.2298e-3$ $(1.55e-2)-$	**$3.9340e-1$ $(6.84e-2)\approx$**	$3.8285e-1$ $(7.56e-2)$
ZDT3	$1.6382e-1$ $(7.15e-2)-$	$2.8712e-2$ $(2.66e-2)-$	$5.9179e-1$ $(1.64e-2)-$	$5.7426e-1$ $(5.53e-2)-$	$2.6975e-1$ $(7.27e-2)-$	$5.9368e-1$ $(2.29e-2)-$	**$6.1053e-1$ $(3.41e-2)$**
ZDT4	$2.6716e-1$ $(2.13e-2)-$	$2.5397e-1$ $(3.13e-1)-$	$3.3190e-1$ $(1.69e-1)-$	$1.8519e-1$ $(1.06e-1)-$	$2.3942e-1$ $(4.32e-1)-$	$1.8471e-1$ $(1.61e-1)-$	**$6.7842e-1$ $(6.31e-2)$**
ZDT6	$4.6442e-3$ $(2.35e-2)-$	$2.8565e-1$ $(1.24e-1)-$	$3.3494e-1$ $(2.92e-2)-$	$2.8767e-1$ $(2.73e-2)-$	**$4.9508e-1$ $(5.70e-2)+$**	$1.9287e-1$ $(5.83e-2)-$	$3.6261e-1$ $(6.91e-2)$
DTLZ1	$1.3921e-2$ $(3.38e-2)-$	$1.5705e-2$ $(6.51e-2)-$	$2.7222e-1$ $(3.27e-1)-$	$4.7545e-1$ $(3.39e-1)\approx$	$3.4441e-1$ $(3.60e-1)\approx$	$3.5922e-1$ $(3.13e-1)\approx$	**$3.6029e-1$ $(1.75e-1)$**
DTLZ2	$4.1179e-1$ $(3.59e-2)-$	$4.8566e-1$ $(7.57e-3)-$	$5.3269e-1$ $(4.35e-3)-$	$5.5494e-1$ $(8.02e-4)\approx$	$5.1427e-1$ $(3.59e-3)-$	**$5.5875e-1$ $(6.24e-4)\approx$**	$5.5705e-1$ $(8.67e-4)$
DTLZ3	$1.5432e-3$ $(5.46e-3)-$	$1.1516e-3$ $(6.31e-3)-$	$2.3794e-3$ $(3.49e-3)-$	$3.0192e-3$ $(2.48e-2)-$	$3.4178e-3$ $(4.52e-3)-$	$4.648e-2$ $(5.47e-2)-$	**$5.3376e-2$ $(1.41e-1)$**
DTLZ4	$3.2572e-1$ $(7.33e-2)-$	$4.8929e-1$ $(1.31e-2)-$	$5.1060e-1$ $(1.16e-3)-$	$3.7188e-1$ $(1.48e-1)-$	$4.8696e-1$ $(2.18e-2)-$	$4.9001e-1$ $(1.19e-1)-$	**$5.5714e-1$ $(8.64e-2)$**
DTLZ5	$1.3942e-1$ $(3.00e-2)-$	$1.9551e-1$ $(6.68e-4)-$	$1.9864e-1$ $(2.35e-4)-$	$1.8239e-1$ $(1.12e-3)-$	$1.9338e-1$ $(4.09e-4)-$	$1.9315e-1$ $(9.44e-4)-$	**$1.9902e-1$ $(1.91e-4)$**
DTLZ6	$2.2663e-3$ $(1.24e-2)-$	$2.1342e-2$ $(5.14e-2)-$	$1.8606e-1$ $(5.06e-2)-$	$1.5778e-1$ $(5.47e-2)-$	$1.9506e-1$ $(1.39e-4)\approx$	$1.8478e-1$ $(3.49e-2)-$	**$1.9587e-1$ $(7.01e-4)$**
DTLZ7	$4.8404e-4$ $(1.22e-3)-$	$4.6002e-6$ $(2.52e-5)-$	**$2.7270e-1$ $(9.81e-3)+$**	$2.2911e-1$ $(1.19e-2)-$	$6.0591e-2$ $(3.07e-2)-$	$2.3749e-1$ $(1.17e-2)-$	$2.4786e-1$ $(7.04e-3)$
$-/+/\approx$	12/0/0	12/0/0	10/1/1	10/0/2	9/1/2	9/0/3	

on more challenging test problems such as DTLZ5 and DTLZ6, the MOEA-UCML algorithm markedly outperforms the other algorithms, obtaining IGD results at the level of 10^{-3}, while the other algorithms yield IGD results at the 10^{-2} level. This evidence showcases the superior global search capability of the MOEA-UCML algorithm.

From Table 2, MOEA-UCML performs well on most of the tested functions for ZDT and DTLZ. On ZDT1, ZDT4, and DTLZ7, MOEA-UCML's results are significantly better than all other algorithms. However, on ZDT6, MOEA-UCML slightly underperforms compared to MOEA/D-D-DE, although the difference is insignificant. Specifically, on ZDT1, MOEA-UCML's HV results are significantly better than all other algorithms. On ZDT4, MOEA-UCML results are substantially ahead of the other algorithms by a wide margin. On DTLZ7, although MOEA-UCML does not lead by much, it still has a clear advantage over most of the algorithms. On the other hand, on some test functions, MOEA-UCML could perform better. For example, on ZDT6, the HV of MOEA-UCML is slightly lower than that of MOEA/D-D-DE. However, MOEA-UCML performs better than or equal to other algorithms in 12 out of 12 test functions, demonstrating its efficiency and robustness.

Figures 3 and 4 illustrate the distributions of the Pareto front on the ZDT1 and DTLZ4 problems for various tested algorithms. As depicted in Figure 3, traditional optimization algorithms, except IBEA, demonstrate challenges in converging towards the Pareto front (PF) of the test problems. On the contrary, MOEA-UCML exhibits substantial superiority, effectively generating high-quality candidate solutions, leading to a solution distribution more closely aligned with the genuine Pareto front. Thus, it can effectively converge towards the optimal solution for each objective function. As illustrated in Figure 4, our method similarly yields superior results on the DTLZ4 problem, demonstrating significant superiority at most iterative points. This result validates that the adopted evolutionary strategy could effectively address various problems and frequently delivers optimal performance.

Furthermore, we conducted a performance comparison between our proposed MOEA-UCML and five classic multiobjective optimization algorithms (including CCGDE3, GDE3, HypE, MOEAD, and MOEADDE), as well as the baseline improvement algorithm, NSGAII, on the WFG test functions. As shown in Table 3, we provide the performance results of all algorithms on each test function, with the best results highlighted in bold. As seen in Table 3, among the nine WFG test functions, MOEA-UCML outperforms CCGDE3, GDE3, HypE, MOEA/D, and MOEA/D-DE on the first, third, fourth, sixth, seventh, eighth, and ninth test functions. It is only slightly inferior to NSGAII and MOEA/D algorithms on the WFG2 and WFG5 problems. Similarly, according to Table 4, MOEA-UCML does show superior performance in global search performance, especially in most of the test functions of WFG, where it outperforms or equals the other compared algorithms. This experiment validates that the uniform initialization mechanism staged evolutionary strategy and the neural network mutation strategy of MOEA-UCML enhance the performance of multiobjective immune optimization algorithms.

In order to present a more quantitative comparison of performance among various algorithms, we plotted the final nondominated solution set obtained from solving the WFG8 problem using each algorithm. As depicted in Figure 5, the approximate solutions derived from the MOEA-UCML algorithm demonstrate the characteristics of closely and evenly covering the actual Pareto front of the WFG8 problem. Comparatively, the solution sets obtained by the other six algorithms show a certain degree of deviation from the Pareto front of the WFG8 problem, exhibiting lower convergence and uniformity, thus underperforming in maintaining the diversity of solutions.

To delve deeper into the performance characteristics and versatility of our proposed MOEA-UCML algorithm, we implemented a comprehensive series of tests that encompass ZDT1-4, 6, DTLZ1-7, WFG1-9, and UF3, 8-10 problems. We plot the Pareto front solutions for each problem in Figure 6. It is noteworthy that the MOEA-UCML algorithm exhibits notable superiority in generating uniformly distributed solution sets. Whether in a two-dimensional or three-dimensional problem space, this algorithm efficiently generates solution sets that exhibit uniform distribution along the Pareto front. This characteristic is particularly crucial in multiobjective optimization problems because it ensures that the solution set covers all possible optimal solutions. Furthermore, the uniformity of the set of solutions allows decision makers to choose from a broader range of solution options, providing greater flexibility. In conclusion, this series of test results powerfully demonstrates the marked superiority and high robustness of the MOEA-UCML algorithm in dealing with various types of multiobjective optimization problems, in terms of convergence, diversity, or uniformity of the set of solutions.

4.3.2. Performance Comparison with Three State-of-the-Art Machine Learning Improved Multiobjective Optimization Algorithms. To verify the performance of the MOEA-UCML algorithm, experiments in this section will be carried out with the ZDT, DTLZ, WFG, and UF test functions. Comparing Tables 1, 3, and 5, it can be observed that the performance of traditional multiobjective optimization algorithms improves to varying degrees after the introduction of machine learning algorithms. From the IGD results, MOEA-UCML outperforms MOEA/D-DQN and its two variants, MOEA/D-OP1 and MOEA/D-OP2, in the 18 corresponding test problems. OP1 represents the simulation of binary crossover, while OP2 denotes the crossover operator in MOEA/D-M2M. MOEA-UCML only performs worse than MOEA/D-OP1 in the ZDT4 and WFG1 problems, is less efficient than MOEA/D-DQN in the WFG4-5 and UF8-9 problems, and slightly inferior to MOEA/D-OP2 in the UF3 problem.

Figure 7 reveals the comparison results for the machine learning-enhanced algorithm (MOEA/D-DQN) and our MOEA-UCML algorithm on the ZDT1 problem. Upon observation, although both algorithms perform similarly in approaching the true Pareto front of the ZDT1 problem, MOEA-UCML exhibits a more distinct advantage in the uniform distribution of nondominated solutions. This result

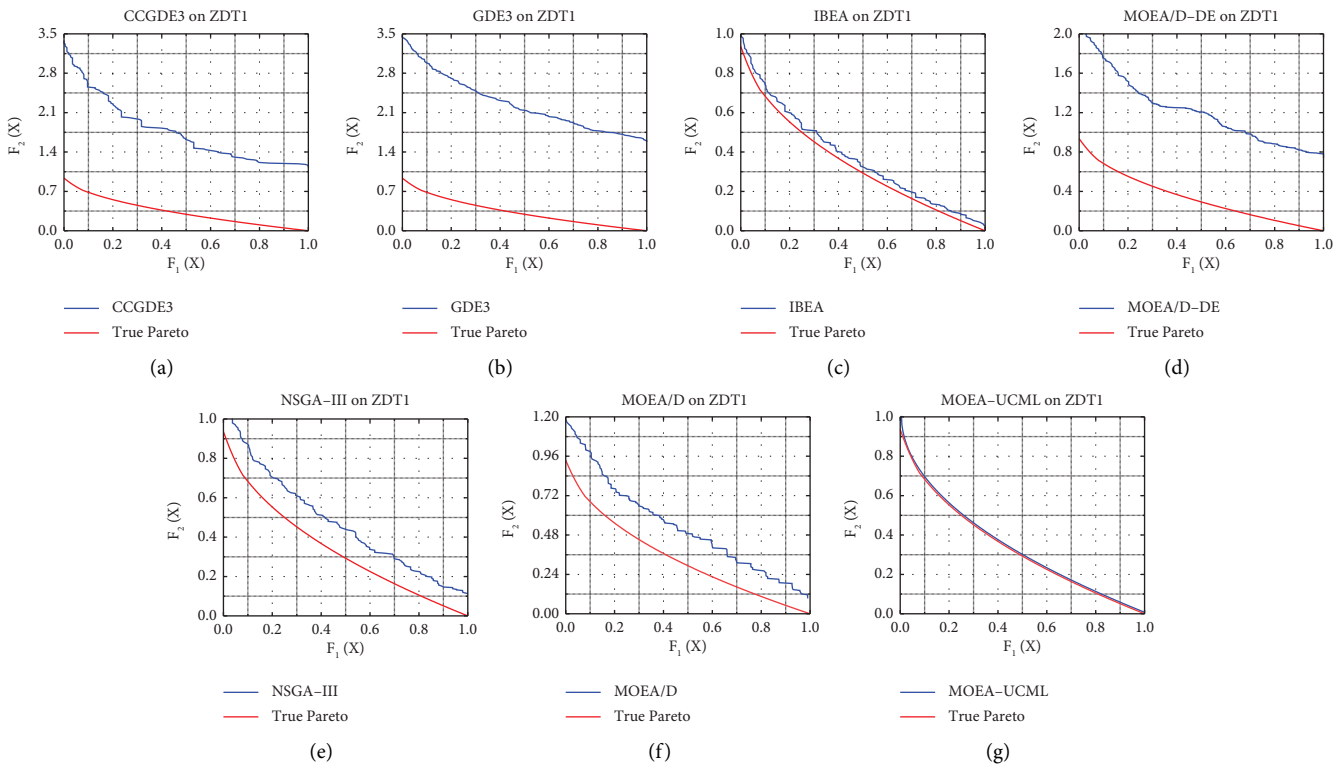

Figure 3: Comparison of Pareto frontier distributions among algorithms under the ZDT1 problem: CCGDE3 (a), GDE3 (b), IBEA (c), MOEA/D (d), MOEA/D-DE (e), NSGAIII (f), and MOEA-UCML (g).

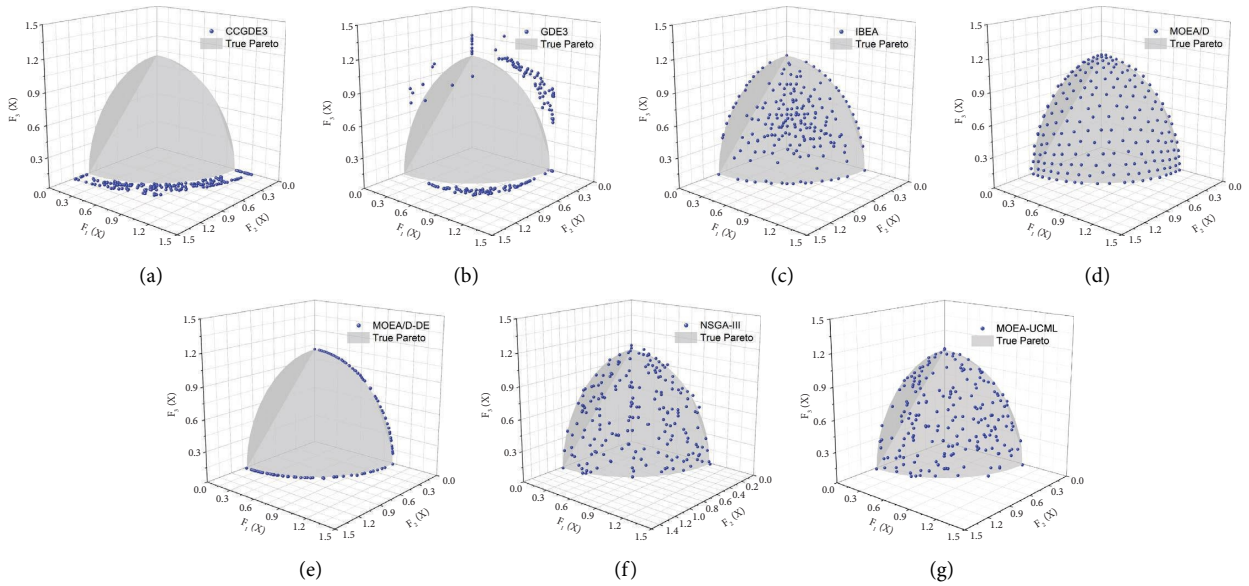

Figure 4: Comparison of Pareto frontier distributions among algorithms under the DTLZ4 problem: CCGDE3 (a), GDE3 (b), IBEA (c), MOEA/D (d), MOEA/D-DE (e), NSGAIII (f), and MOEA-UCML (g).

underscores the unique value of MOEA-UCML, i.e., its capability to maintain excellent uniformity of the solution set while searching for high-quality solutions. Therefore, despite the exemplary performance of MOEA/D-DQN on specific performance metrics, the significant advantage of MOEA-UCML in generating uniformly distributed nondominated solutions renders it more practically valuable when dealing with multiobjective optimization problems.

4.3.3. Performance Comparison with Two Improved Multi-objective Optimization Algorithms with Advanced Objective Initialization. To verify the population initialization

Table 3: The IGD comparison results of MOEA-UCML with six traditional multiobjective optimization algorithms on the WFG problem sets.

Problem	CCGDE3	GDE3	HypE	MOEA/D	MOEA/D-DE	NSGAII	MOEA-UCML
WFG1	$1.7336e+0\ (1.20e-1)-$	$1.6986e+0\ (4.30e-2)-$	$1.0326e+0\ (8.60e-2)-$	$6.8356e-1\ (9.47e-2)-$	$1.4705e+0\ (5.63e-2)-$	$5.6150e-1\ (6.58e-2)-$	**$4.9263e-1\ (7.96e-2)$**
WFG2	$4.1103e-1\ (5.95e-2)-$	$2.8494e-1\ (1.35e-2)-$	$2.8632e-1\ (7.96e-3)-$	$3.2244e-1\ (6.32e-2)-$	$3.6277e-1\ (2.33e-2)-$	**$2.2001e-1\ (7.28e-3)\approx$**	$2.2624e-1\ (1.00e-2)$
WFG3	$4.0099e-1\ (5.65e-2)-$	$3.5928e-1\ (2.58e-2)-$	$3.1067e-2\ (3.03e-3)+$	$3.1744e-1\ (9.29e-2)-$	$2.7073e-1\ (2.60e-2)-$	$1.2346e-1\ (1.67e-2)-$	**$1.1915e-1\ (1.60e-2)$**
WFG4	$3.5323e-1\ (2.43e-2)-$	$3.5333e-1\ (1.43e-2)-$	$2.6839e-1\ (1.12e-2)+$	$2.9518e-1\ (1.27e-2)-$	$3.9702e-1\ (1.02e-2)-$	$2.8109e-1\ (1.10e-2)-$	**$2.7556e-1\ (9.68e-3)$**
WFG5	$3.8142e-1\ (1.81e-2)-$	$2.7836e-1\ (1.06e-2)\approx$	$3.0093e-1\ (1.38e-2)-$	**$2.7299e-1\ (7.25e-3)+$**	$3.3684e-1\ (3.72e-3)-$	$2.8423e-1\ (8.75e-3)-$	$2.7339e-1\ (9.38e-3)$
WFG6	$4.1809e-1\ (4.43e-2)-$	$3.1899e-1\ (2.07e-2)-$	$2.9994e-1\ (1.40e-2)-$	$2.9172e-1\ (2.74e-2)-$	$3.2113e-1\ (4.16e-2)-$	$2.6845e-1\ (1.29e-2)-$	**$2.6029e-1\ (1.91e-2)$**
WFG7	$3.3425e-1\ (2.15e-2)-$	$2.9632e-1\ (9.29e-3)-$	$3.2576e-1\ (1.47e-2)-$	$3.2500e-1\ (4.56e-2)-$	$2.4840e-1\ (6.52e-3)-$	$2.0655e-1\ (6.52e-3)-$	**$2.0147e-1\ (5.02e-3)$**
WFG8	$4.5156e-1\ (1.95e-2)-$	$4.0750e-1\ (1.19e-2)-$	$3.3085e-1\ (1.00e-2)-$	$3.1976e-1\ (1.93e-2)\approx$	$3.5242e-1\ (1.27e-2)-$	$3.1619e-1\ (6.25e-3)\approx$	**$3.1162e-1\ (7.98e-3)$**
WFG9	$3.0796e-1\ (4.15e-2)-$	$3.4566e-1\ (8.10e-3)-$	$2.8404e-1\ (1.48e-2)-$	$2.8369e-1\ (7.06e-2)-$	$2.2949e-1\ (4.69e-3)-$	$2.0442e-1\ (6.31e-3)\approx$	**$2.0381e-1\ (7.84e-3)$**
$-/+/\approx$	9/0/0	8/0/1	7/2/0	7/1/1	9/0/0	6/0/3	

Table 4: The HV comparison results of MOEA-UCML with six traditional multiobjective optimization algorithms on the WFG problem sets.

Problem	CCGDE3	GDE3	HypE	MOEA/D	MOEA/D-DE	NSGAII	MOEA-UCML
WFG1	$1.4113e-1$ $(5.19e-2)-$	$1.6485e-1$ $(2.22e-2)-$	$5.5204e-1$ $(4.59e-2)-$	$5.9891e-1$ $(5.04e-2)-$	$2.8526e-1$ $(1.98e-2)-$	$6.9271e-1$ $(4.19e-2)-$	**$7.5707e-1$ $(4.14e-2)$**
WFG2	$7.7919e-1$ $(2.93e-2)-$	$8.4679e-1$ $(8.81e-3)-$	$9.0441e-1$ $(8.07e-3)-$	$8.2780e-1$ $(4.31e-2)-$	$8.5648e-1$ $(1.22e-2)-$	**$9.3513e-1$ $(3.74e-3)$** $\approx$	$9.2813e-1$ $(3.58e-3)$
WFG3	$2.5375e-1$ $(2.77e-2)-$	$2.7007e-1$ $(1.24e-2)-$	$3.7515e-1$ $(9.33e-3)-$	$2.8869e-1$ $(3.21e-2)-$	$2.8538e-1$ $(2.92e-2)-$	$3.7690e-1$ $(8.64e-3)-$	**$4.0733e-1$ $(1.90e-3)$**
WFG4	$4.4494e-1$ $(1.34e-2)-$	$4.4011e-1$ $(5.80e-3)-$	$5.0989e-1$ $(4.13e-3)-$	$4.9072e-1$ $(7.59e-3)-$	$4.4422e-1$ $(7.32e-3)-$	$5.0249e-1$ $(6.15e-3)-$	**$5.5398e-1$ $(1.51e-3)$**
WFG5	$3.9788e-1$ $(1.75e-2)-$	$4.7175e-1$ $(9.35e-3)-$	$4.8268e-1$ $(3.50e-3)-$	**$5.2866e-1$ $(6.85e-3)+$**	$4.5770e-1$ $(4.23e-3)-$	$4.8188e-1$ $(4.52e-3)-$	$5.1690e-1$ $(1.03e-3)$
WFG6	$3.3601e-1$ $(1.77e-2)-$	$4.1341e-1$ $(1.70e-2)-$	$4.5474e-1$ $(1.94e-2)-$	$4.3797e-1$ $(1.88e-2)-$	$3.8472e-1$ $(2.20e-2)-$	$4.5300e-1$ $(1.51e-2)-$	**$5.0050e-1$ $(1.54e-2)$**
WFG7	$4.0162e-1$ $(1.47e-2)-$	$4.2670e-1$ $(1.02e-2)-$	$5.1806e-1$ $(6.11e-3)-$	$4.3715e-1$ $(2.70e-2)-$	$4.5957e-1$ $(1.27e-2)-$	$5.1079e-1$ $(5.54e-3)-$	**$5.5370e-1$ $(1.13e-3)$**
WFG8	$3.3275e-1$ $(7.78e-3)-$	$3.5328e-1$ $(9.36e-3)-$	$4.2703e-1$ $(7.52e-3)-$	$4.0907e-1$ $(2.17e-2)-$	$3.4300e-1$ $(1.42e-2)-$	$4.2525e-1$ $(5.96e-3)$ $\approx$	**$4.5667e-1$ $(2.53e-3)$**
WFG9	$3.9728e-1$ $(2.87e-2)-$	$3.8579e-1$ $(1.84e-2)-$	$4.8036e-1$ $(2.32e-2)-$	$4.1361e-1$ $(3.73e-2)-$	$4.5710e-1$ $(2.20e-2)-$	$4.7746e-1$ $(2.49e-2)-$	**$5.3328e-1$ $(4.48e-3)$**
$-/+/\approx$	9/0/0	9/0/0	9/0/0	8/1/0	9/0/0	7/0/2	

The bold values indicate the optimal performance results of all algorithms for each test function.

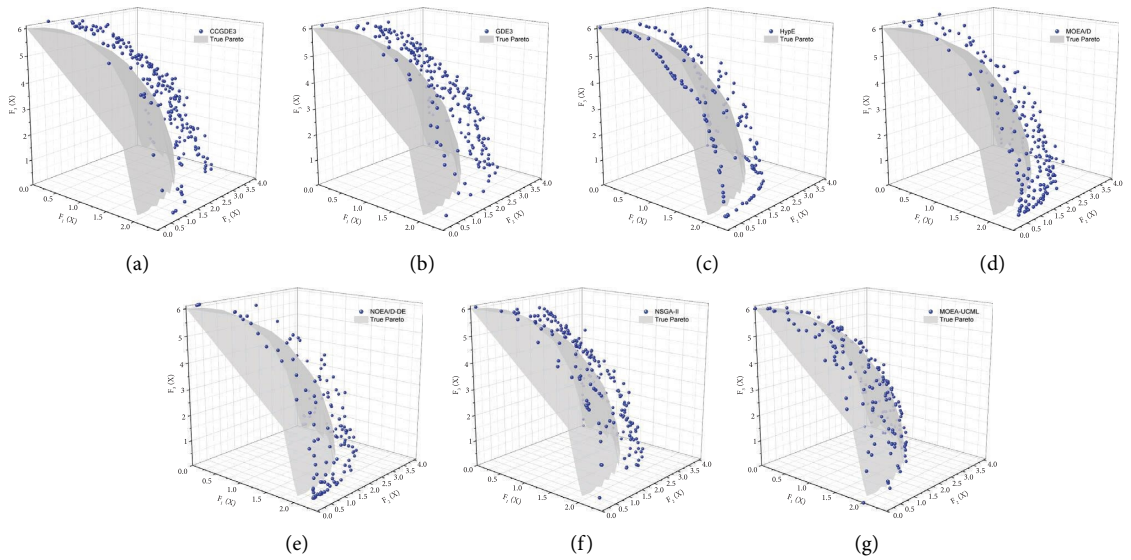

Figure 5: Comparison of Pareto frontier distributions among algorithms under the WFG8 problem: CCGDE3 (a), GDE3 (b), HypE (c), MOEA/D (d), MOEA/D-DE (e), NSGAII (f), and MOEA-UCML (g).

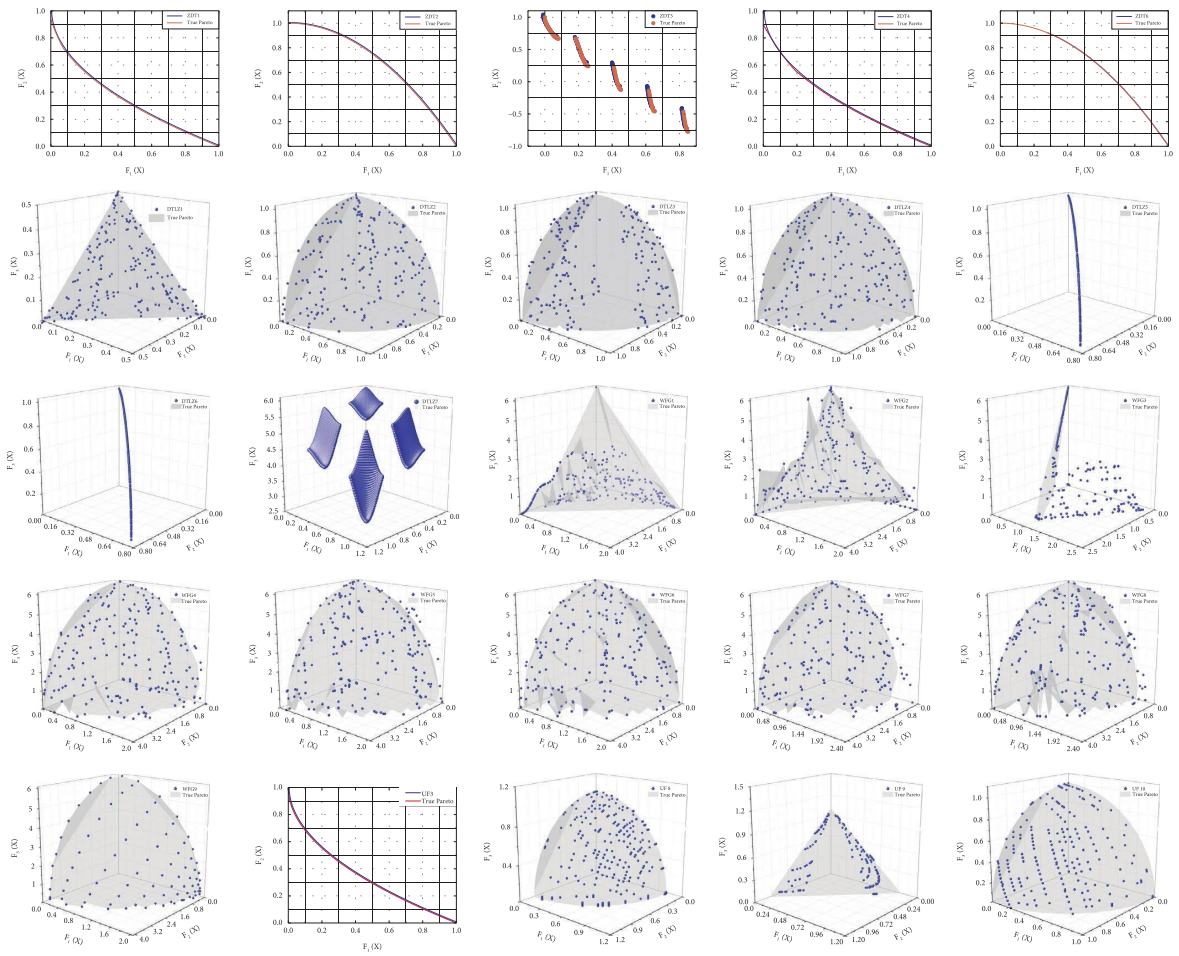

Figure 6: MOEA-UCML undominated solution Pareto frontier distribution matrix on problem sets ZDT1-4, 6, DTLZ1-7, WFG1-9, and UF3, 8-10.

Table 5: The IGD comparison results of MOEA-UCML with three machine learning improved multiobjective optimization algorithms on ZDT, DTLZ, WFG, and UF problem sets.

Problem	MOEA/D-OP1	MOEA/D-OP2	MOEA/D-DQN	MOEA-UCML
ZDT3	4.3307e − 2 (2.58e − 2)−	2.2010e − 1 (6.31e − 2)−	1.7122e − 2 (5.58e − 3)−	**1.6742e − 2 (5.00e − 3)**
ZDT4	**1.2382e − 1 (6.48e − 2)+**	6.2584e − 1 (3.03e − 1)−	2 5453e − 1 (9.70e − 2)−	6.0438e − 2 (8.63e − 2)
DTLZ5	3.3019e − 2 (4.74e − 4)−	3.0277e − 2 (1.11e − 3)−	2.6538e − 2 (5.78e − 4)−	**5.8972e − 3 (2.45e − 4)**
DTLZ6	3.3856e − 2 (4.15e − 5)−	3.3576e − 2 (1.34e − 4)−	2.9008e − 2 (4.72e − 5)−	**5.8490e − 3 (2.97e − 4)**
DTLZ7	1.5409e − 1 (1.57e − 3)−	1.5373e − 1 (2.81e − 3)−	1.1006e − 1 (3.56e − 4) ≈	**1.0100e − 1 (5.32e − 2)**
WFG1	**2.5873e − 1 (2.10e − 2)+**	6.8505e − 1 (1.46e − 1)−	7.1675e − 1 (8.33e − 2)−	4.9263e − 1 (7.96e − 2)
WFG2	2.3869e − 1 (2.12e − 2)−	2.9602e − 1 (2.49e − 2)−	2.0742e − 1 (1.96e − 2)−	**2.2624e − 1 (1.00e − 2)**
WFG3	1.5931e − 1 (4.76e − 3)−	1.6717e − 1 (1.43e − 2)−	1.2825e − 1 (4.77e − 3)−	**1.1915e − 1 (1.60e − 2)**
WFG4	2.6389e − 1 (6.64e − 3)+	3.4070e − 1 (1.65e − 2)−	**2.3703e − 1 (5.38e − 3)+**	2.7556e − 1 (9.68e − 3)
WFG5	2.5097e − 1 (3.38e − 3)+	2.7192e − 1 (7.79e − 3) ≈	**2.2353e − 1 (3.01e − 3)+**	2.7339e − 1 (9.38e − 3)
WFG6	2.9117e − 1 (1.93e − 2)−	3.9338e − 1 (1.15e − 2)−	4.2542e − 1 (4.02e − 2)−	**2.6029e − 1 (1.91e − 2)**
WFG7	2.8538e − 1 (1.65e − 2)−	3.7268e − 1 (2.80e − 2)−	3.6882e − 1 (1.13c-2)−	**2.0147e − 1 (5.02e − 3)**
WFG8	3.3973e − 1 (8.30e − 3)−	5.0260e − 1 (3.83e − 2)−	4.8352e − 1 (5.64e − 2)−	**3.1162e − 1 (7.98e − 3)**
WFG9	3.0634e − 1 (4.95e − 2)−	3.5093e − 1 (2.58e − 2)−	3.5458e − 1 (3.56e − 2)−	**2.0381e − 1 (7.84e − 3)**
UF3	3.4550e − 1 (4.64e − 2)−	**1.5582e − 1 (1.01e − 1)+**	2.1607e − 2 (1.97e − 2)+	3.0769e − 1 (4.36e − 2)
UF8	2.5450e − 1 (1.92e − 1)−	2.3313e − 1 (6.28e − 2)−	**9.3945e − 2 (7.22e − 2)+**	2.0175e − 1 (2.85e − 2)
UF9	2.2444e − 1 (7.01e − 2)−	1.9439e − 1 (1.07e − 2) ≈	**7.7286e − 2 (6.56e − 2)+**	2.0951e − 1 (7.06e − 2)
UF10	7.6045e − 1 (2.01e − 1)−	6.5741e − 1 (8.38e − 2)−	5.3962e − 1 (1.80e − 1)−	**4.6333e − 1 (2.42e − 1)**
−/+/≈	14/4/0	15/1/2	12/5/1	

The bold values indicate the optimal performance results of all algorithms for each test function.

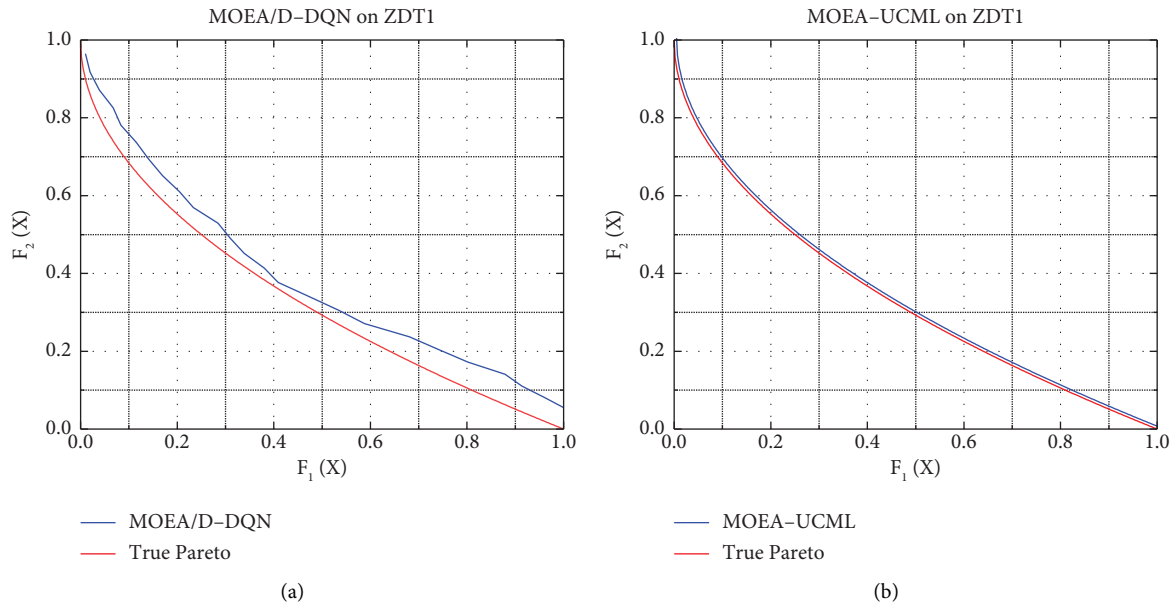

Figure 7: Uniformity of the Pareto frontier distribution for MOEA/D-DQN versus MOEA-UCML on the ZDT1 problem.

performance of the MOEA-UCML algorithm, experiments in this section will be carried out under the DTLZ, WFG, and UF test functions. The algorithm proposed in this paper was compared in terms of performance with two advanced objective initialization-improved multiobjective optimization algorithms, CMOPSO and MOEA/D-AAWN. By comparing Tables 1, 2, and 6, it can be observed that the performance of traditional multiobjective optimization algorithms improves to varying degrees after introducing new population initialization strategy algorithms. From the IGD results, MOEA-UCML outperforms the other two improved algorithms in 9 corresponding test problems. Here, CMOPSO represents a multistep initialization mechanism based on decision variable division, and MOEA/D-AAWN represents an initialization improvement algorithm based on decomposition that adaptively adjusts the weight vector and neighborhood. MOEA-UCML only performs worse than MOEA/D-AAWN in ZDT5 and WFG4-5 problems and is slightly less effective than CMOPSO in the DTLZ6 problem.

We further examined the impact of objective initialization, particularly by comparing the performance of MOEA/D-AAWN and our MOEA-UCML algorithm on the ZDT1 and ZDT2 problems. As shown in Figure 8, regardless of the test problem, the MOEA-UCML algorithm exhibited a more pronounced advantage in approaching the true Pareto front. This implies that our

Table 6: The IGD comparison results of MOEA-UCML with two initialization improved multiobjective optimization algorithms on DTLZ and WFG problem sets.

Problem	CMOPSO	MOEA/D-AAWN	MOEA-UCML
DTLZ4	$1.4430e-1\ (2.8e-1)-$	$4.8771e-1\ (3.5e-1)-$	**$1.0003e-1\ (2.0e-1)$**
DTLZ5	$4.5173e-3\ (3.4e-4)-$	**$3.2439e-3\ (1.2e-4)+$**	$5.8972e-3\ (2.4e-4)$
DTLZ6	**$4.3734e-3\ (6.1e-5)+$**	$1.5703e-1\ (3.0e-1)-$	$5.8490e-3\ (3.0e-4)$
WFG4	$3.8183e-1(3.9e-3)-$	**$2.1752e-1\ (1.4e-2)+$**	$2.7556e-1\ (9.7e-3)$
WFG5	$7.7588e-1\ (1.8e-3)-$	**$2.2549e-1\ (1.2e-2)+$**	$2.7339e-1\ (9.4e-3)$
WFG6	$8.6534e-1\ (1.2e-2)-$	$2.8890e-1\ (2.1e-2)-$	**$2.6029e-1\ (1.9e-2)$**
WFG7	$2.9628e-1\ (2.2e-3)-$	$2.0988e-1\ (5.4e-2)\approx$	**$2.0147e-1\ (5.0e-3)$**
WFG8	$4.0653e-1\ (8.1e-3)-$	$3.1907e-1\ (4.9e-2)\approx$	**$3.1162e-1\ (8.0e-3)$**
WFG9	$7.3108e-1\ (3.9e-2)-$	$2.6916e-1\ (5.1e-2)-$	**$2.0381e-1\ (7.8e-3)$**
$-/+/\approx$	8/1/0	4/3/2	

The bold values indicate the optimal performance results of all algorithms for each test function.

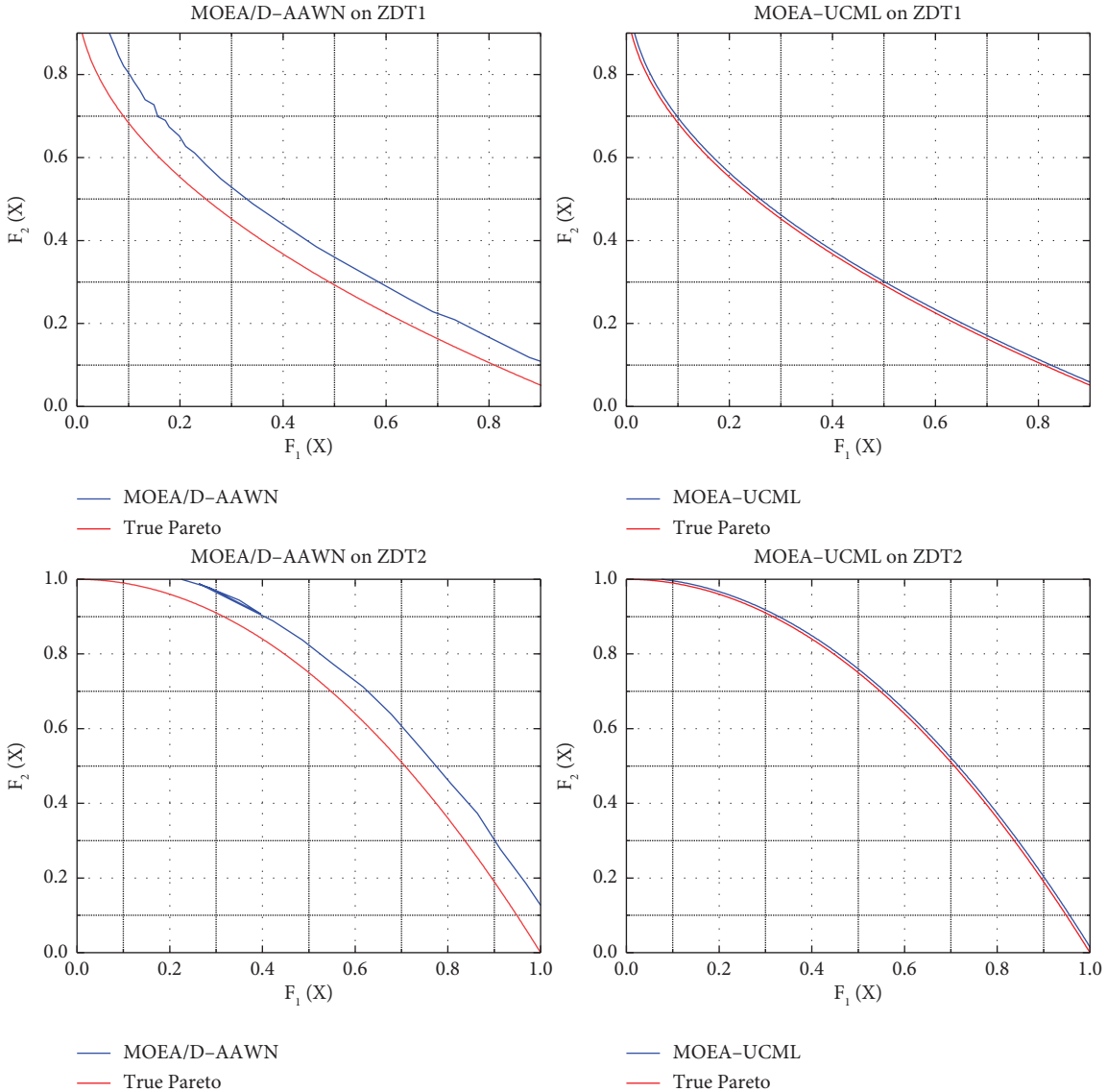

Figure 8: Comparison of Pareto frontier approximation between MOEA/D-AAWN and MOEA-UCML on ZDT1 and ZDT2 problems.

algorithm can effectively find high-quality solutions with a smaller distance to the true Pareto front, demonstrating its superior performance and robustness when handling these multiobjective optimization problems. Furthermore, this also emphasizes the significant impact of objective initialization on solving multiobjective optimization problems, particularly on solution quality and the algorithm's convergence performance.

Table 7: The running time comparison results between MOEA-UCML and three traditional multiobjective optimization algorithms on the DTLZ problem set.

Problem	HypE	MOEAD	MOEADDE	MOEA-UCML
DTLZ1	$5.7786e+1$ $(2.28e+1)-$	$2.5043e+0$ $(2.73e-1)\approx$	$2.1339e+0$ $(1.72e-1)\approx$	**$2.2106e+0$ $(8.67e+0)$**
DTLZ2	$3.4190e+2$ $(2.00e+1)-$	$2.5045e+0$ $(2.44e-1)-$	$2.1360e+0$ $(1.52e-1)-$	**$1.7138e+0$ $(1.01e+0)$**
DTLZ3	$1.7855e+1$ $(1.96e+0)-$	$2.5450e+0$ $(2.05e-1)-$	$2.1945e+0$ $(1.59e-1)-$	**$1.6234e+0$ $(1.16e+0)$**
DTLZ4	$2.7614e+2$ $(4.10e+1)-$	$2.5628e+0$ $(2.26e-1)-$	$2.1925e+0$ $(1.43e-1)-$	**$1.6232e+0$ $(5.71e-1)$**
DTLZ5	$2.2814e+2$ $(1.73e+1)-$	$2.5606e+0$ $(2.58e-1)-$	$2.2153e+0$ $(1.20e-1)-$	**$1.5807e+0$ $(7.11e-1)$**
DTLZ6	$4.2194e+2$ $(3.51e+1)-$	$2.7896e+0$ $(3.49e-1)-$	$2.2638e+0$ $(1.59e-1)-$	**$1.5644e+0$ $(9.17e-1)$**
DTLZ7	$9.6128e+1$ $(1.33e+1)-$	$2.5970e+0$ $(2.32e-1)-$	$2.2420e+0$ $(1.95e-1)-$	**$1.5553e+0$ $(8.01e-1)$**
$-/+/\approx$	7/0/0	6/0/1	6/0/1	

The bold values indicate the optimal performance results of all algorithms for each test function.

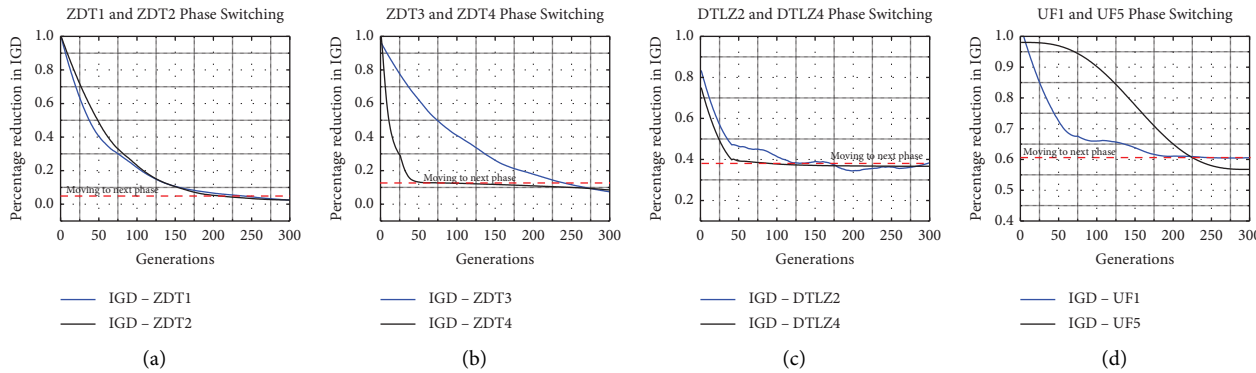

Figure 9: IGD real-time tracking indicator threshold setting and MOEA-UCML phase-switching effect.

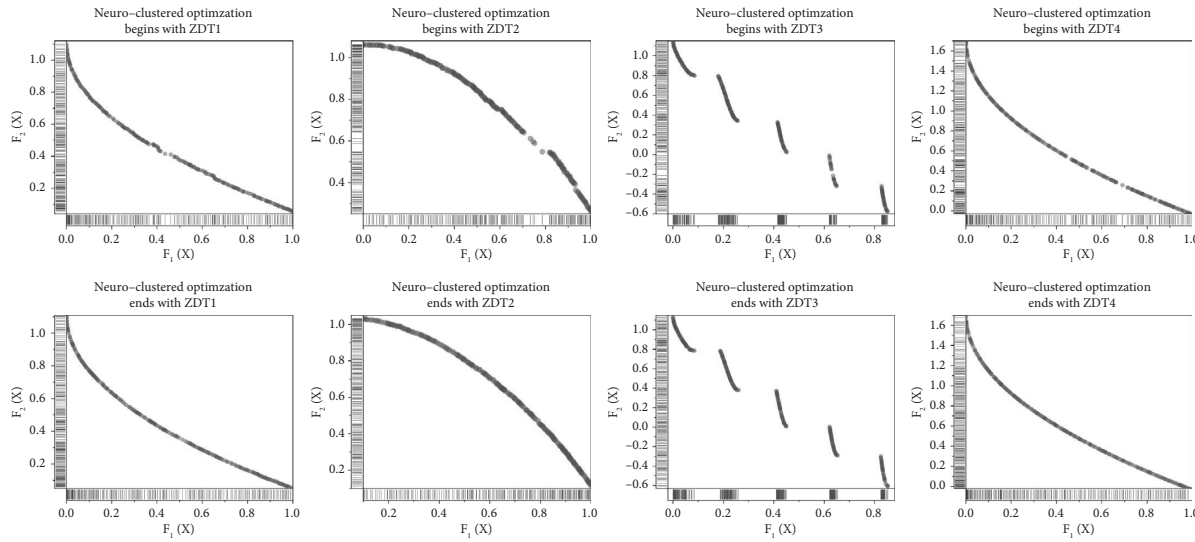

Figure 10: MOEA-UCML comparison of Pareto frontiers before and after neuro-clustered optimization on the ZDT1-4 problem.

4.3.4. Runtime Evaluation of MOEA-UCML Multiobjective Optimization Algorithm. Finally, to assess the efficiency of the MOEA-UCML algorithm, we performed a detailed analysis of its runtime and performed performance comparisons on the ZDT and DTLZ problem sets. We kept all parameter settings consistent to ensure the fairness and reliability of the experiments. As can be seen from the results in Table 7, the MOEA-UCML algorithm demonstrated superior performance on multiple test problems. This is mainly due to the phase-switching mechanism we have designed, which reduces the algorithm's runtime to some extent. Overall, compared to single-strategy algorithms, MOEA-UCML exhibited a lower time cost on most test problems, further validating the effectiveness of our proposed phase-switching mechanism. Furthermore, as shown in Figure 9, we set different threshold values for the IGD real-time tracking metrics for different problems, such as approximately 90% for the ZDT problem set, around 60% for the DTLZ problem set, and about 40% for the UF problem set. Therefore, these experimental results strongly confirm

the significant effect of the phase-switching mechanism in the MOEA-UCML algorithm, which will undoubtedly enhance the overall performance of Pareto-based multiobjective evolutionary algorithms.

4.3.5. Further Discussion of MOEA-UCML. Based on the aforementioned experimental results, our MOEA-UCML algorithm outperforms eight traditional multiobjective optimization algorithms, three machine learning improved algorithms, and two algorithms enhanced with initialization in terms of performance. However, we conducted further comparative experiments to provide a more thorough validation of the effectiveness of neuro-clustered optimization in MOEA-UCML. Specifically, we delineated the Pareto front of MOEA-UCML before and after fine-tuning on ZDT1-4 problems, offering a more intuitive illustration of the impact of the neuro-clustered optimization phase. As depicted in Figure 10, compared to before fine-tuning, the solutions found by the MOEA-UCML algorithm after fine-tuning are more uniformly distributed along the Pareto front. This shows that the fine-tuning process can significantly enhance the uniformity of the solution distribution, further confirming the effectiveness of our neural network neuro-clustered optimization strategy.

5. Conclusion

In this study, we have proposed a novel multiobjective optimization algorithm—MOEA-UCML, employing a multistage optimization approach that combines a uniform initialization strategy, a self-organizing map optimization strategy, an IGD-driven transition strategy, and a mutation strategy. While MOEA-UCML exhibits high-quality solutions and search efficiency in multiobjective optimization problems, it also has certain limitations. First, using a neural network-guided mutation strategy in the algorithm, which requires training the neural network in each generation, could potentially increase computational complexity. Second, the algorithm's performance is heavily based on the settings of parameters such as population size, the number of layers in the neural network, and the number of clusters in K-means clustering. These settings require appropriate adjustment in practical applications.

In summary, the MOEA-UCML algorithm still harbors excellent potential in dealing with multiobjective optimization problems. In future work, we plan to further optimize the performance of this algorithm, such as exploring more effective mutation strategies, improving the neural network training process to reduce computational complexity, and seeking more suitable parameter adjustment strategies. Moreover, we plan to apply this algorithm to a broader range of fields, including bioinformatics, computer science, and manufacturing, to address complex multiobjective optimization problems in practice.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon reasonable request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported in part by the Youth Natural Science Foundation of Sichuan (grant no. 2023NSFSC1395), in part by the National Natural Science Foundation of China (grant no. 62101358), in part by the China Postdoctoral Science Foundation (no. 2020M683345), in part by the Joint Innovation Fund of Sichuan University, and Nuclear Power Institute of China (grant no. HG2022143).

References

[1] A. Zhou, B. Y. Qu, H. Li, S. Z. Zhao, P. N. Suganthan, and Q. Zhang, "Multiobjective evolutionary algorithms: a survey of the state of the art," *Swarm and Evolutionary Computation*, vol. 1, no. 1, pp. 32–49, 2011.

[2] W. Jiang, S. You, J. Zhan, X. Wang, H. Lei, and D. Adhikari, "Query-efficient generation of adversarial examples for defensive DNNs via multi-objective optimization," *IEEE Transactions on Evolutionary Computation*, vol. 27, no. 4, pp. 832–847, 2023.

[3] K. Wang, J. Wang, B. Zeng, and H. Lu, "An integrated power load point-interval forecasting system based on information entropy and multi-objective optimization," *Applied Energy*, vol. 314, Article ID 118938, 2022.

[4] R. Khalili, A. Khaledi, M. Marzband, A. F. Nematollahi, B. Vahidi, and P. Siano, "Robust multi-objective optimization for the Iranian electricity market considering green hydrogen and analyzing the performance of different demand response programs," *Applied Energy*, vol. 334, Article ID 120737, 2023.

[5] Y. Bi, B. Xue, P. Mesejo, S. Cagnoni, and M. Zhang, "A survey on evolutionary computation for computer vision and image analysis: past, present, and future trends," *IEEE Transactions on Evolutionary Computation*, vol. 27, no. 1, pp. 5–25, 2023.

[6] P. Huang, P. He, S. Tian et al., "A ViT-AMC network with adaptive model fusion and multiobjective optimization for interpretable laryngeal tumor grading from histopathological images," *IEEE Transactions on Medical Imaging*, vol. 42, no. 1, pp. 15–28, 2023.

[7] Q. Zhang and H. Li, "MOEA/D: a multiobjective evolutionary algorithm based on decomposition," *IEEE Transactions on Evolutionary Computation*, vol. 11, no. 6, pp. 712–731, 2007.

[8] Q. Zhu, Q. Zhang, and Q. Lin, "A constrained multiobjective evolutionary algorithm with detect-and-escape strategy," *IEEE Transactions on Evolutionary Computation*, vol. 24, no. 5, pp. 938–947, 2020.

[9] L. R. de Farias and A. F. Araújo, "A decomposition-based many-objective evolutionary algorithm updating weights when required," *Swarm and Evolutionary Computation*, vol. 68, Article ID 100980, 2022.

[10] T. Takagi, K. Takadama, and H. Sato, "Weight vector arrangement using virtual objective vectors in decomposition-based MOEA," in *Proceedings of the 2021 IEEE Congress on Evolutionary Computation (CEC)*, pp. 1462–1469, IEEE, Kraków, Poland, June 2021.

[11] K. Deb, A. Pratap, S. Agarwal, and T. A. M. T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: nsga-II," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 2, pp. 182–197, 2002.

[12] L. Pan, W. Xu, L. Li, C. He, and R. Cheng, "Adaptive simulated binary crossover for rotated multi-objective optimization," *Swarm and Evolutionary Computation*, vol. 60, Article ID 100759, 2021.

[13] A. López Jaimes, C. A. Coello Coello, H. Aguirre, and K. Tanaka, "Objective space partitioning using conflict information for solving many-objective problems," *Information Sciences*, vol. 268, pp. 305–327, 2014.

[14] Y. Tian, R. Cheng, X. Zhang, Y. Su, and Y. Jin, "A strengthened dominance relation considering convergence and diversity for evolutionary many-objective optimization," *IEEE Transactions on Evolutionary Computation*, vol. 23, no. 2, pp. 331–345, 2019.

[15] K. Li, K. Deb, Q. Zhang, and S. Kwong, "An evolutionary many-objective optimization algorithm based on dominance and decomposition," *IEEE Transactions on Evolutionary Computation*, vol. 19, no. 5, pp. 694–716, 2015.

[16] L. M. Antonio and C. A. C. Coello, "Use of cooperative coevolution for solving large scale multiobjective optimization problems," in *Proceedings of the 2013 IEEE Congress on Evolutionary Computation*, pp. 2758–2765, IEEE, Cancun, Mexico, June 2013.

[17] L. Yan, W. Qi, J. Liang et al., "Inter-individual correlation and dimension based dual learning for dynamic multi-objective optimization," *IEEE Transactions on Evolutionary Computation*, vol. 1, 2023.

[18] E. Zitzler, M. Laumanns, and L. Thiele, "SPEA2: improving the strength Pareto evolutionary algorithm," *TIK report*, vol. 103, 2001.

[19] M. Li, S. Yang, and X. Liu, "Shift-based density estimation for Pareto-based algorithms in many-objective optimization," *IEEE Transactions on Evolutionary Computation*, vol. 18, no. 3, pp. 348–365, 2014.

[20] F. Zhao, W. Lei, W. Ma, Y. Liu, and C. Zhang, "An improved SPEA2 algorithm with adaptive selection of evolutionary operators scheme for multiobjective optimization problems," *Mathematical Problems in Engineering*, vol. 2016, Article ID 8010346, 20 pages, 2016.

[21] F. He, K. Shen, L. Guan, and M. Jiang, "Research on energy-saving scheduling of a forging stock charging furnace based on an improved SPEA2 algorithm," *Sustainability*, vol. 9, no. 11, p. 2154, 2017.

[22] S. Elsayed, R. Sarker, and C. A. Coello Coello, "Sequence-based deterministic initialization for evolutionary algorithms," *IEEE Transactions on Cybernetics*, vol. 47, no. 9, pp. 2911–2923, 2017.

[23] A. Deniz and H. E. Kiziloz, "On initial population generation in feature subset selection," *Expert Systems with Applications*, vol. 137, pp. 11–21, 2019.

[24] W. Huang and W. Zhang, "Multi-objective optimization based on an adaptive competitive swarm optimizer," *Information Sciences*, vol. 583, pp. 266–287, 2022.

[25] P. Wang, Y. Ma, and M. Wang, "A dynamic multi-objective optimization evolutionary algorithm based on particle swarm prediction strategy and prediction adjustment strategy," *Swarm and Evolutionary Computation*, vol. 75, Article ID 101164, 2022.

[26] Z. Lv, L. Wang, Z. Han, J. Zhao, and W. Wang, "Surrogate-assisted particle swarm optimization algorithm with Pareto active learning for expensive multi-objective optimization," *IEEE/CAA Journal of Automatica Sinica*, vol. 6, no. 3, pp. 838–849, 2019.

[27] F. Zou, G. G. Yen, L. Tang, and C. Wang, "A reinforcement learning approach for dynamic multi-objective optimization," *Information Sciences*, vol. 546, pp. 815–834, 2021.

[28] S. Li, S. Yang, Y. Wang, W. Yue, and J. Qiao, "A modular neural network-based population prediction strategy for evolutionary dynamic multi-objective optimization," *Swarm and Evolutionary Computation*, vol. 62, Article ID 100829, 2021.

[29] W. Liu, L. Chen, X. Hao, W. Zhou, X. Cao, and F. Xie, "Offspring regeneration method based on bi-level sampling for large-scale evolutionary multi-objective optimization," *Swarm and Evolutionary Computation*, vol. 75, Article ID 101152, 2022.

[30] Y. Tian, X. Li, H. Ma, X. Zhang, K. C. Tan, and Y. Jin, "Deep reinforcement learning based adaptive operator selection for evolutionary multi-objective optimization," *IEEE Transactions on Emerging Topics in Computational Intelligence*, vol. 7, no. 4, pp. 1051–1064, 2023.

[31] X. Cai, Y. Xiao, M. Li, H. Hu, H. Ishibuchi, and X. Li, "A grid-based inverted generational distance for multi/many-objective optimization," *IEEE Transactions on Evolutionary Computation*, vol. 25, no. 1, pp. 21–34, 2021.

[32] W. Chen, H. Ishibuchi, and K. Shang, "Modified distance-based subset selection for evolutionary multi-objective optimization algorithms," in *Proceedings of the 2020 IEEE Congress on Evolutionary Computation (CEC)*, pp. 1–8, IEEE, Glasgow, UK, July 2020.

[33] H. Han, Y. Liu, Y. Hou, and J. Qiao, "Multi-modal multi-objective particle swarm optimization with self-adjusting strategy," *Information Sciences*, vol. 629, pp. 580–598, 2023.

[34] Z. Song, H. Wang, B. Xue, M. Zhang, and Y. Jin, "Balancing objective optimization and constraint satisfaction in expensive constrained evolutionary multi-objective optimization," *IEEE Transactions on Evolutionary Computation*, vol. 1, 2023.

[35] E. Zitzler and S. Künzli, "Indicator-based selection in multiobjective search," in *International Conference on Parallel Problem Solving from Nature*, pp. 832–842, Springer Berlin Heidelberg, Berlin, Heidelberg, 2004.

[36] J. Bader and E. Zitzler, "HypE: an algorithm for fast hypervolume-based many-objective optimization," *Evolutionary Computation*, vol. 19, no. 1, pp. 45–76, 2011.

[37] K. Deb and H. Jain, "An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part I: solving problems with box constraints," *IEEE Transactions on Evolutionary Computation*, vol. 18, no. 4, pp. 577–601, 2014.

[38] Q. Gu, Y. Liu, L. Chen, and N. Xiong, "An improved competitive particle swarm optimization for many-objective optimization problems," *Expert Systems with Applications*, vol. 189, Article ID 116118, 2022.

[39] Q. Zhao, Y. Guo, X. Yao, and D. Gong, "Decomposition-based multi-objective optimization algorithms with adaptively adjusting weight vectors and neighborhoods," *IEEE Transactions on Evolutionary Computation*, vol. 27, no. 5, pp. 1485–1497, 2023.