mathematics

Article

A Synergistic MOEA Algorithm with GANs for Complex Data Analysis

Weihua Qian [1,†], Hang Xu [2,†], Houjin Chen [1], Lvqing Yang [1,*], Yuanguo Lin [1], Rui Xu [3], Mulan Yang [4] and Minghong Liao [1]

[1] School of Informatics, Xiamen University, Xiamen 361000, China; clydeqian@stu.xmu.edu.cn (W.Q.); chenhoujin1248@163.com (H.C.); xdlyg@stu.xmu.edu.cn (Y.L.); liao@xmu.edu.cn (M.L.)
[2] School of Mechanical, Electrical & Information Engineering, Putian University, Putian 351100, China; hangxu520@hotmail.com
[3] College of Computer and Data Science, Fuzhou University, Fuzhou 350108, China; xurui.ryan.chn@gmail.com
[4] School of Economics and Finance, Xi'an Jiaotong University, Xi'an 710061, China; 15280284728@163.com
* Correspondence: lqyang@xmu.edu.cn
† These authors contributed equally to this work.

Abstract: The multi-objective evolutionary algorithm optimization (MOEA) is a challenging but critical approach for tackling complex data analysis problems. However, prevailing MOEAs often rely on single strategies to obtain optimal solutions, leading to concerns such as premature convergence and insufficient population diversity, particularly in high-dimensional data scenarios. In this paper, we propose a novel adversarial population generation algorithm, APG-SMOEA, which synergistically combines the benefits of MOEAs and Generative Adversarial Networks (GANs) to address these limitations. In order to balance the efficiency and quality of offspring selection, we introduce an adaptive population entropy strategy, which includes control parameters based on population entropy and a learning pool for storing and retrieving optimal solutions. Additionally, we attempt to alleviate the training complexity and model collapse problems common in GANs with APG-SMOEA. Experimental results on benchmarks demonstrate that the proposed algorithm is superior to the existing algorithms in terms of solution quality and diversity of low-dimensional or high-dimensional complex data.

Keywords: multi-objective optimization; evolutionary algorithm; generative adversarial network; complex data analysis

MSC: 68W50

check for updates

Citation: Qian, W.; Xu, H.; Chen, H.; Yang, L.; Lin, Y.; Xu, R.; Yang, M.; Liao, M. A Synergistic MOEA Algorithm with GANs for Complex Data Analysis. *Mathematics* **2024**, *12*, 175. https://doi.org/10.3390/math12020175

Academic Editor: Ioannis G. Tsoulos

Received: 9 December 2023
Revised: 2 January 2024
Accepted: 4 January 2024
Published: 5 January 2024

1. Introduction

In recent years, complex data analysis has become an essential research field due to the increasing and diversified data generated by various sources [1], such as traffic logistics, commercial transactions, financial investment, and healthcare services [2,3]. One of the most significant and prevalent challenges in analyzing complex data include the multi-objective optimization problems (MOPs) [4], as optimizing one objective often leads to the deterioration of others. Traditional optimization methods, such as gradient-based methods, can only optimize one objective at a time, which may not be able to address MOPs effectively. In this context, multi-objective evolutionary algorithms (MOEAs) [5], which incorporate evolutionary algorithms (EAs) into MOPs, have emerged as a promising solution.

MOEAs are designed to find a set of solutions that balance multiple objectives and achieve the best trade-offs among them, which are known as Pareto optimal solutions. Specifically, a solution is Pareto optimal if no other solution exists that can improve at least one goal without compromising any of the others. Moreover, the set of all Pareto optimal solutions is called the Pareto front or Pareto set. To explore the solution space, MOEAs generate a population of candidate solutions and then apply genetic operators such as mutation, crossover, and selection to evolve better solutions over generations. Typical MOEAs can be categorized into four types: Domination-based MOEAs, such as NSGA-II [6] and NSGA-III [7], utilize the Pareto-dominance principle to achieve convergence and maintain diversity, but may suffer from the problem of slow convergence. Indicator-based MOEAs, such as SMS-EMOA [8], use quality indicators to evaluate approximation sets and impose a total order among related solutions, but with high computational complexity. Decomposition-based MOEAs (known as MOEA/D), such as MOEA/D-AWA [9], decompose MOPs into several scalar sub-problems and solve them simultaneously. In MOEA/HD [10], sub-problems are layered into different hierarchies, and the search directions of lower-hierarchy sub-problems are adaptively adjusted, according to the higher-hierarchy search results. However, they may face challenges in generating high-quality and diverse offspring for complex high-dimensional data. Hybrid MOEAs, such as Hybrid-MOEA/D-I [11], combine different algorithms to solve complex MOPs, but they can be challenging to design and implement.

Compared to domination- and indicator-based MOEAs, MOEA/D stands out for its unique decomposition strategy, which enables it to be more easily combined with other optimization methods to potentially improve its performance. To further enhance the performance of MOEA/D, substantial research [12–18] has been conducted recently. In particular, Qian et al. [13] proposed MOEA/D-AEE with a joint coefficient for adaptive exploration and development. It has been proven to be effective in achieving improved sparse distribution and global search capabilities compared with several classical MOEA/D algorithms [19]. However, like many MOEA/D algorithms, it still faces limitations in generating high-quality and diverse offspring, especially in dealing with complex high-dimensional data. These limitations can potentially hinder the performance of MOEA/D-AEE in solving complex multi-objective optimization problems.

Generative Adversarial Networks (GANs) show tremendous potential in generating high-quality synthetic data. The classical GANs consist of generator network and discriminator network. Specifically, the generator network takes a random noise vector as input and produces synthetic data, while the discriminator network tries to distinguish between synthetic and real data. During training, GANs leverage an adversarial loss function that simultaneously minimizes the generator's loss and maximizes the discriminator's loss, leading to the generation of high-quality synthetic data. The effectiveness of GANs has been demonstrated in various applications, such as image and text generation [20], data augmentation [21], and style transfer [22].

In the context of MOEA, GANs can be used to generate diverse and high-quality offspring populations, further enhancing the optimization process. Compared to traditional MOEA methods, the synergistic approach based on the MOEA and GAN has several benefits. First, the GAN can produce diverse and high-quality offspring populations by learning the data distribution through adversarial training, thus preventing premature convergence and enhancing the exploration of the search space. Second, the GAN can overcome the limitations of traditional MOEA methods in generating diverse offspring populations, which can lead to more optimal solutions. Third, the synergistic approach can adapt to different dimensions of data, which improves the performance of the MOEA algorithm in complex data.

Despite the synergistic approach of combining EAs and GANs for MOEA optimization having shown great potential, it also has its limitations. One major challenge is how to balance the trade-off between exploration and exploitation when selecting offspring from the GAN-generated and MOEA-generated population. Additionally, training a GAN can be challenging, as it typically requires a large amount of data. Moreover, GAN-generated populations may suffer from the problem of mode collapse, resulting in poor quality and lack of diversity in offspring populations. To address these challenges, this paper proposes a novel adversarial population generation-based algorithm, namely, *Synergistic MOEA Algorithm with Adversarial Population Generation* (APG-SMOEA), that synergistically combines the strengths of EAs and GANs. In summary, this paper makes the following main contributions:

1. This paper introduces the GANs into the MOEA/D algorithm. This new algorithm combines the advantages of both MOEAs and GANs. It can utilize the simple framework of multi-objective evolution algorithms and the advantages of GANs in generating samples and high-dimensional data analysis.
2. This paper introduces an adaptive entropy control strategy that includes a population entropy-based control parameter. This strategy allows us to balance the exploration and exploitation trade-off by controlling the diversity of the GAN-generated population.Effectively avoid premature convergence and population diversity issues in APG-SMOEA algorithm.
3. This paper introduces a quality hybrid memory pool to store and retrieve optimal solutions. This strategy not only improves the algorithm's ability to maintain diversity and avoid premature convergence, but also alleviates the problem of high quality data desire in GAN training. By storing the best solutions from previous generations, we can prevent the algorithm from becoming stuck in a local optima, while also allowing for the exploration of new regions of the search space.
4. Our algorithm effectively mitigates the common problems of training complexity and model collapse in GANs. Experimental results on benchmarks demonstrate that our proposed algorithm outperforms existing methods in terms of solution quality and diversity on complex data with low or high dimensions.

2. Related Work

2.1. MOEAs

As the classical algorithms that have been popular for many years, MOEAs [23,24] are an effective way to solve complex MOPs. Generally, MOEAs can be classified into the following four classes, according to their characteristics [25]:

1. Domination-based MOEAs [26], such as NSGA-II [6,27] and NSGA-III [7], usually achieve the convergence by means of Pareto-dominance principle, while maintaining the diversity of solutions via explicit diversity preservation. However, such MOEAs suffer from the slow convergence since Pareto-dominance principle has the low selection mechanism.Domination-based MOEAs can be described in following equation.
   The population size is N, the objective function value of the i-th individual in the population is f_i, then the domination set of this individual is defined by Equation (1).
   $$D(i) = \{j \mid j \in \{1, 2, \ldots, N\} \wedge f_j < f_i\} \tag{1}$$
   where $D(i)$ is the dominated set of individual i, which is the index set of all individuals dominated by individual i. The i-th individual domination is defined as follows in Equation (2).
   $$\pi_i = \frac{|D(i)|}{N} \tag{2}$$
   where $|D(i)|$ is the number of elements in the dominated set of individual i, the domination degree of individual i. Finally, based on the domination degree, the population is selected to generate the next generation. There can be many specific selection methods, such as tournament selection, sorting selection, etc.
2. Indicator-based MOEAs [28–32] adopt quality indicators, which define the selection mechanism as a function to evaluate approximation sets. For example, in the S-Metric Selection Evolutionary Multi-objective Algorithm (SMS-EMOA) [8], the dominated volume of an approximation set is calculated by the hypervolume indicator. Based on this, a total order among related solutions is imposed. An example of a weighted sum index calculation formula is shown in Equation (3).

$$f(x) = \sum_{i=1}^{n} w_i f_i(x) \tag{3}$$

where $f(x)$ is the index value of solution x, n is the number of indices, w_i is the weight of the i-th index, and $f_i(x)$ is the function value of the i-th index.

3. Decomposition-based MOEAs, also called MOEA/D [33–35], decompose a MOP into several scalar sub-problems by employing the decomposition strategy. Thus, such sub-problems are solved simultaneously by optimizing the solution of each sub-problem. To cope with the limitations of MOEA/D, some research works [14] have proposed new methods for weight vector generation. For example, the MOEA/D-AWA [9] employs an adaptive weight vector adjustment strategy to achieve better uniformity of solutions to the target MOPs. The MOEA/D's general equation is as follows in Equation (4).

$$F(x) = [f_1(x), f_2(x), \ldots, f_n(x)]^T \tag{4}$$

where $F(x)$ is the objective function value vector of solution x, $f_i(x)$ is the function value of the i-th objective function, and n is the number of objective functions.
In the MOEA/D algorithm, the multi-objective optimization problem needs to be decomposed into a series of single-objective optimization sub-problems, each corresponding to a reference point. For each reference point r_i, a weight vector $w = [w_1, w_2, ..., w_n]^T$ needs to be defined, where $w_i > 0$, and the weighted function value $f_i(x) = w^T * F(x)$ needs to be calculated. Then, define a sub-problem L_i: minimize $f_i(x)$ subject to $x \in X$. Use an evolutionary algorithm or other optimization method to solve each sub-problem, and obtain a set of non-dominated solutions S_i. Merge all non-dominated solutions into a solution set S, and select a set of non-dominated solutions from S as the final result. Repeat the above steps until the termination condition is met.

4. Hybrid MOEAs [36–38] often combine different algorithms in a unified framework to solve complex MOPs. For example, the Hybrid-MOEA/D-I [11] combines a differential evolutionary algorithm with a genetic algorithm to improve the performance, which effectively optimizes MOPs in wireless sensor networks. Moreover, in the multi-objective particle swarm optimization (MOPSO) [39], a novel particle swarm optimization approach can solve a class of mean-variance in portfolio selection problems, and an adaptive ranking procedure further improves the performance of the proposed algorithm.

2.2. GAN

Theoretically, GAN is a generative model that uses unsupervised learning to generate synthetic or fake data [40]. It consists of two models: the generator network and the discriminator network [41]. The generator network captures the distribution of training data and generates realistic data, whereas the discriminator network is a classifier that differentiates generated data from training data as accurately as possible [42]. In other words, to deal with the shortcomings of other generative algorithms, GAN introduces adversarial learning to distinguish the generated data (i.e., synthetic or fake data) from true data.

In general, the objective function, in original minmax cases, of GAN can be defined as follows [43].

$$\min_{G} \max_{D} V(D, G) = E_{x \sim P_g(x)}[\log(D(x))] + E_{z \sim P_z(z)}[\log(1 - D(G(z)))] \tag{5}$$

where G denotes the generator, D denotes the discriminator, $P_g(x)$ is a generator distribution over data x, $\log(D(x))$ denotes the cross-entropy value between $[1 \quad 0]^T$ and $[D(x) \quad 1 - D(x)]^T$, $P_z(z)$ is a prior on the input noise variable z, and $\log(1 - D(G(z)))$ denotes the cross-entropy value between $[0 \quad 1]^T$ and $[D(G(z)) \quad 1 - D(G(z))]^T$.

However, the generator G usually works poorly during the early learning process. To this end, we may maximize $\log D(G(z))$ instead of minimizing $\log(1 - D(G(z)))$ to reduce the cost for training the generator [42]:

$$\begin{aligned} J^{(G)} &= E_{z \sim P_z(z)}[-\log(D(G(z)))] \\ &= E_{x \sim P_g(x)}[-\log(D(x))] \end{aligned} \tag{6}$$

Similarly, we also can minimize the following function by the maximum likelihood, assuming that the discriminator $G(z)$ is optimal, then [44]

$$\begin{aligned} J^{(G)} &= E_{z \sim P_z(z)}[-exp(\sigma^{-1}(D(G(z))))] \\ &= E_{z \sim P_z(z)}\left[\frac{-(D(G(z))}{1 - D(G(z))}\right] \end{aligned} \tag{7}$$

where σ is the logistic sigmoid function.

In recent years, GAN has become a hot research topic and has gained remarkable success in various fields [45,46], such as text-to-image synthesis [47], human pose synthesis [48], handwritten font generation [49], medical applications [50], face frontal view generation [51], text-to-image synthesis [52], and anime characters generation [53]. In summary, the applications of GAN algorithms can be classified into two categories: The first is computer vision [54], including image applications [55], object detection [56], video applications [57], and texture synthesis [58]. The second is data engineering, including spatio-temporal data [59], music generation [60], data synthesis [61], and sequential data processing [62]. In this work, to address the problem of high quality data desire in GAN training, we propose a quality hybrid memory pool to store and retrieve optimal solutions.

3. The Proposed Algorithm

In this section, we first provide the definition for the solved problems. Then we propose the specific algorithm framework. Finally, we detail three major innovations.

3.1. Problem Definition

In MOPs, optimizing a single objective can come at the cost of other objectives, while achieving multiple optimal objectives may require finding an optimal constraint relationship between them [63]. The classical portfolio optimization, for example, has to satisfy both the highest return and the lowest risk. In reality, return and risk are often constrained, as achieving higher returns may lead to higher risks, while lowering risks may result in lower returns. Similar classical problems exist in the fields of recommendation systems, traffic logistics, path planning, and financial investment. Therefore, the MOPs have been a more important research direction in the field of scientific research, and the classical MOEA algorithm has proposed a better solution. Consider the weights of each objective and find a set of optimal solution sets that satisfy the problem constraints, i.e., Pareto solution sets. In [13], the authors proposed the MOEA/D-AEE algorithm, incorporating dynamic search and joint development techniques. This algorithm addresses the issue of self-optimization in low- and medium-dimensional data and enhances the algorithm's self-adaptive capacity and global search capability. However, an effective solution for large-scale MOPs with a large number of objectives and high complexity has not yet been found. Our team proposes the APG-SMOEA algorithm, which is based on the MOEA/D-AEE, to tackle MOPs with complex data. To evaluate the effectiveness of our proposed algorithm, this paper uses a financial dataset from the OR library, which contains data with high dimensions up to 225. The dataset addresses a problem that can be formulated mathematically as follows:

$$\max_{\omega} RETURN = \sum_{i=1}^{n} r_i \omega_i \tag{8}$$

$$\max_{\omega} RISK = \sum_{i=1}^{n}\sum_{j=1}^{n}\sigma_{ij}\omega_i\omega_j \tag{9}$$

subject to,

$$\sum_{i=1}^{n}\omega_i = 1(0 \le \omega_i \le 1) \tag{10}$$

where $\omega = (w_1, w_2, ..., w_n)$ is a vector representing the invest ratios of n assets. r_i is the return rate of i-th asset, and σ_{ij} is co-variance between the return rated of i-th asset and j-th asset.

The APG-SMOEA algorithm aims to solve the issues of early convergence and population diversity in the MOEA algorithm. These are the three key challenges:

- Challenge 1: How to synergize GAN networks?
- Challenge 2: How to solve the problem of model collapse arising from the introduction of GAN?
- Challenge 3: How to introduce an adaptive control strategy?

These are the solutions:

1. Fusion of traditional arithmetic with GANs.
2. Increase the generation probability of GAN networks when the population diversity decreases.
3. Population entropy determines the proportion of mixed operators.

3.2. APG-SMOEA Framework

This paper uses high-dimensional portfolio data and specifically uses the portfolio optimization problem to represent a MOPs with complex data. The MOP problem is converted into a matrix vector form, where the corresponding solution is represented by a weight vector consisting of floating-point numbers between 0 and 1.

The variational formulation of the MOEA/D-Lévy algorithm [19] is shown below in Equation (11)

$$y(\varepsilon) = \varepsilon x^i + \alpha_0\left(x^i - x^j\right) \oplus Levy(\beta) \tag{11}$$

where x^i and x^j are the two parents used for progeny reproduction, y is the generated progeny, $\oplus$ is the method of multiplying item by item, α_0 is the scale factor used, and $Levy(\beta)$ is the vector generated by mantegna algorithm (MA).

The Lévy flight mutation operator utilized in this study exhibits distinct differences in its application to distinct individuals. It is worth noting that the scaling factor utilized in the Lévy flight mutation is derived from a heavy-tailed distribution. Furthermore, Lévy flight mutation typically necessitates only two parents. The Lévy flight mutation formula is outlined in the following Equation (12):

$$\text{Levy}(\beta) \sim \frac{u}{|v|^{\frac{1}{\beta}}} \tag{12}$$

$$u \sim N\left(0, \sigma_u^2\right), v \sim N\left(0, \sigma_v^2\right) \tag{13}$$

$$\sigma_v = \left[\frac{\Gamma(1+\beta) \bullet \sin\left(\frac{\pi\beta}{2}\right)}{\Gamma\left(1+\frac{\beta}{2}\right) \bullet \beta \cdot 2^{\beta-\frac{1}{2}}}\right]^{\frac{1}{\beta}}, \sigma_v = 1 \tag{14}$$

where β is a parameter of the Lévy mutation algorithm. u is a random variable with a mean of 0 and a variance of σ_u^2. v is a random variable with a mean of 0 and a variance of σ_v^2. The expressions for σ_v and σ_u are provided in Equation (14). Γ is a function.

Due to the limited global search and adaptive control capabilities of the Lévy flight algorithm, the quality of offspring generated in the later stages of iteration often suffers.

To address these issues, MOEA/D-AEE introduces a dynamic coefficient to dynamically explore the solution space, thereby enhancing the Lévy flight algorithm's performance [13]. This is achieved by choosing a random number ε between 0 and 1, and then one of the parents is selected at random with ε. If ε is less than σ, the parent is selected from the neighbors. Otherwise, the parent is selected from the initial population. In addition, the algorithm selects another parent in the initial population with probability $1 - \varepsilon$. The formulation of this algorithm [13] is shown in (15) below.

$$y(\varepsilon) = \varepsilon x^i + \alpha_0(1-\varepsilon)\left(x^i - x^j\right) \oplus Levy(\beta) \tag{15}$$

The MOEA/D-AEE algorithm [13] has a strong ability for global search and adaptation. The performance metrics of MOEA/D-AEE are notably superior to those of classical multi-objective optimization algorithms like NSGA-II [6], MOEA/D-Lévy [19], and MOEA/D-DEM.

The APG-SMOEA algorithm is further optimized based on MOEA/D-AEE algorithm. Figure 1 shows the process and Algorithm A1 (in Appendix A.1) provides the pseudocode, where t is the number of loop iterations, i is used to traverse each individual in the population, and σ uses the diversity maintenance strategy in [33]. b is the neighbor of the current individual. α controls the probability of GAN network generation. M and V are the $[return, risk]$ of the individuals in the population. $update\_count$ is the individual update counter. n_r is the upper limit of update neighbors. In the generation operation of offspring, the dynamic parameter α controlled by the population entropy determines the generation ratio between the polynomial variation and GANs.

The following are the detailed steps of the APG-SMOEA algorithm:

1. Read the information of the dataset and set the parameters of the problem. This includes statistical information on the number of assets, returns and risks, and correlation information (covariance matrix). It is also necessary to set the initial parameters of the algorithm, such as population size, number of neighbors, etc. (Algorithm A1 line 1)
2. Define the neighbors, generate the initial population, and calculate the individual $[return, risk]$ and Pareto front extreme value points. (Algorithm A1 line 2–3)
3. Construct the GAN network. (Algorithm A1 line 4 and Algorithm A2)
4. Loop iterations; each iteration is a complete search process, including selection, variation, crossover, replacement, and other operations. (Algorithm A1 line 5–27)
5. Each iteration traverses each individual in the population. This step generates new children by performing mutation and crossover operations on each individual. (Algorithm A1 line 6–26)
6. Determine the mating pool of the offspring generated by the variation operator. The generated random number is compared with σ. If the random number is greater than σ, the whole population is used as the mating pool. Otherwise, the neighbors of individuals are used as the mating pool. (Algorithm A1 line 7)
7. Determine the way of generation of offspring. Compare the generated random number with α. If the random number is greater than α, choose the polynomial variation generated offspring determined by Equation (2); otherwise, choose the way to generate offspring by GAN network training (see Algorithm A2 for related pseudocode, in Appendix A.2). Compute the $[return, risk]$ of the offspring. (Algorithm A1 line 8–12)
8. Update the extreme value points of the Pareto front, compute the reference points and weight vectors to decide whether to replace the parent, and then update the population. (Algorithm A1 line 13–15)
9. Set the counter, traverse the neighbors, update the population if the children are better than the parents, and exit the loop when the traversal ends or the counter reaches the upper limit. (Algorithm A1 line 16–24)

10. Calculate the population entropy according to Equation (19), update the alpha, and record the new population entropy and alpha (see Algorithm A3 for the relevant pseudo-code, in Appendix A.3). (Algorithm A1 line 25 and Algorithm A3)

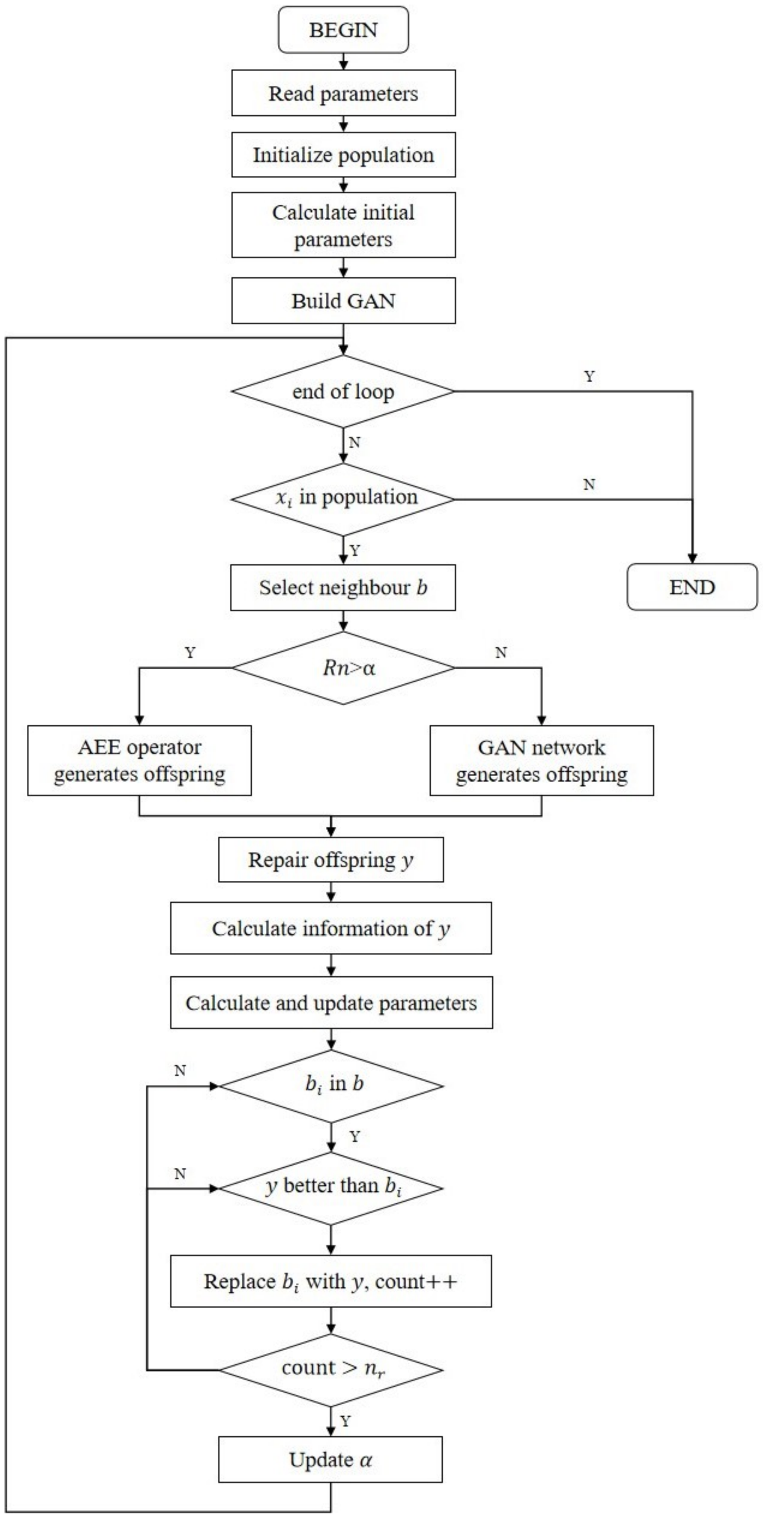

Figure 1. Algorithm flow chart.

The children generated by the hybrid operator (AEE operator and GAN networks), especially the GAN network, may not satisfy the constraints, so the repair operation is needed for the children. The repair steps are as follows:

1. Iterate through the weights of all assets in the child generation and set the negative numbers to 0.
2. Sum the weights of all children s assets .
3. If $s \neq 0$, all the weight vectors are scaled so that their sum is 1. Otherwise, the children satisfying the constraint are generated randomly.

The time complexity of the APG-SMOEA algorithm mainly manifests during the generation process of descendants using Generative Adversarial Networks (GANs). First, we use a fast non-dominated sorting algorithm to select training samples, where the number of objectives is denoted as M, and the population size is N. The time complexity for a single individual undergoing sorting is $O(MN)$ (requiring comparisons with $N-1$ individuals, each involving comparison of M objectives). The initial round involving N individuals entails a time complexity of $O(MN^2)$. In the worst-case scenario, where only one individual replaces the current non-dominated solution in each round, $O(N)$ rounds of sorting are necessary, with each round consuming $O(N)$ time complexity, and the worst-case time complexity of the other rounds of sorting is $O(N^2)$. Thus, the time complexity of the fast non-dominated sorting algorithm is $O(MN^2) = O(MN^2) + O(N^2)$.

Furthermore, the training of GANs also contributes to the time complexity. The generator comprises two fully connected layers and two hidden layers, while the discriminator consists of one fully connected layer and one hidden layer. During the training process, the mean and covariance matrix of real samples are computed, followed by the generation of samples Z from a noise distribution for training the discriminator D. GAN networks undergo training every 100 epochs.

The extensive computations involved in the fast non-dominated sorting and GAN training processes consume a considerable amount of time. Consequently, compared to multi-objective optimization algorithms based on conventional methods, the time performance of this algorithm is significantly reduced.

3.3. Introduction of GANs

Classical multi-objective optimization algorithms have problems with premature convergence and insufficient population diversity. Although many algorithms have been proposed to solve these problems, their performance on high-dimensional complex data varies widely [41]. The initial population of the APG-SMOEA algorithm is randomly generated. In the early iterations, there are few high-quality solutions in the population. If we use these few solutions to train the GAN network, it is easy to encounter premature convergence/insufficient population diversity/local optimal problems. Therefore, the APG-SMOEA algorithm uses population entropy strategy to coordinate the MOEA/D-AEE algorithm and GAN network to generate offspring, so that more high-quality solutions are generated by MOEA/D-AEE in the early iterations and more offspring are generated by GAN network in the later iterations.This strategy addresses the issues of premature convergence and insufficient population diversity.

Studies have shown that when dealing with MOPs for complex high-dimensional data, GAN networks have significant advantages. In addition, GAN networks can generate data by adversarial training, which is very similar to the way multi-objective optimization algorithms generate new individuals by parental crossover variation. The effectiveness of GAN networks in MOPs was demonstrated in conjunction with the literature [12]. The specific pseudo-code of the algorithm is shown in Algorithm A2 (in Appendix A.2). For the method of generating offspring by GAN network, if the offspring need to be generated, the population is firstly sorted by fast non-dominated ordering, the top ten individuals are screened out as the real samples (non-dominated solutions) for training, and the other dominated solutions are used as false samples, which are inputted into the GAN network to be trained, at the same time generating the offspring. At the end of that, the set of solutions generated by the GAN network is sorted by fast non-dominated ordering, and the top twenty individuals are screened out to be used in the replacement of the population. The loss function of the discriminator in the GAN network used in this experiment is presented in Equation (16).

$$\max_{\mathrm{D}} V(D) = E_{r\sim P_f}[\log(D(r))] + E_{f\sim P_f}[\log(1 - D(f))] + E_{z\sim P_z}[\log(1 - D(G(z)))] \quad (16)$$

where $D(r), D(f), D(G(z))$ denote the output of the discriminator, respectively; the distributions of the real and fake datasets are denoted as P_r and P_f. The positive samples (good individuals in the population), negative samples (other individuals in the population), and samples generated by the generator are the inputs.

The following steps are necessary for generating progeny by the GAN network:

1. Calculate the covariance matrix according to the mean vector μ. Obtain the sum of the real samples according to Equation (17),

$$\sum = \frac{\sum_{i=1}^{\lfloor N/2 \rfloor}(r_i - \mu)(r_i - \mu)^T}{\lfloor N/2 \rfloor - 1}, \text{ where } \mu = \frac{\sum_{i=1}^{\lfloor N/2 \rfloor} r_i}{\lfloor N/2 \rfloor} \quad (17)$$

   Here, N is the population size, and r_i is the i-th member of the positive sample.
2. Sample the D-dimensional vector x. Generate a D-dimensional vector y', which satisfies the multivariate normal distribution according to Equation (18).

$$y' = \frac{\exp\left(-1/2(\mathrm{x} - \mu)^{\mathrm{T}}\Sigma^{-1}(\mathrm{x} - \mu)\right)}{\sqrt{(2\pi)^{\mathrm{D}}|\Sigma|}} \quad (18)$$

   where μ is the mean vector of the positive samples. $\sum$ is the sample covariance of the positive samples. D is the dimensionality of the feature.
3. The generator generates children based on the upper boundary l and the lower boundary u of the decision space, as shown in Equation (19).

$$y = G(y')(u - l) + 1 \quad (19)$$

   where y is the candidate solution generated by the GAN networks. l and μ are the upper and lower boundaries of the decision space.

3.4. Solving the Model Collapse Problem

In the APG-SMOEA algorithm, the initial population is generated by randomization, and the number of good solutions in the population may be relatively small in the early iterations. Retraining the GAN network using the solutions it generates can lead to the problem of model collapse. This paper proposes a new hybrid operator to solve these problems. This approach fuses the GAN network with the MOEA/D-AEE operator, while using adaptive parameters to control the probability of generating offspring from the GAN network training (adaptive parameters will be described in Section 3.5). The specific formula is shown in Equation (20) below:

$$y = \begin{cases} \varepsilon x^y + \alpha_0(1-\varepsilon)\left(x^i - x^j\right) \oplus Levy(\beta), Rn > \alpha \\ G\left(\frac{\exp\left(-1/2(x-\mu)^T \Sigma^{-1}(x-\mu)\right)}{\sqrt{(2\pi)^D|\Sigma|}}\right)(u - l) + l, Rn < \alpha \end{cases} \quad (20)$$

where Rn is between 0 and 1. α is an adaptive parameter determined by the population entropy.

The use of hybrid operators allows for the complete training of GAN networks, enabling them to search for more regions and optimize further in terms of convergence. The hybrid operator approach combines the benefits of GAN networks in diversity and reduces the risk of model collapse.

3.5. Adaptive Entropy Control Strategy

Multi-objective optimization often suffers from a common issue where the population diversity is large in the early stage, but small in the later stage, leading to the risk of getting stuck in a local optimum. The idea of solving this problem is generally to increase the diversity of late stage populations through specific methods. The MOEA/D-AEE algorithm introduces joint coefficients for a good optimization of this typical problem. However, it is not effective in increasing the later population diversity. This paper improves the previous work by using the APG-SMOEA algorithm and a population entropy adaptive control strategy, which aligns better with the characteristics of multi-objective optimization. By adjusting the GAN network generation probability, we can use the GAN network to increase population diversity in the early iteration and decrease it in the late iteration, which can be beneficial for optimizing the population using the GAN network. Based on this idea, the local optimum problem is greatly solved. (Refer to Algorithm A3 for the specific pseudo-code, in Appendix A.3).

In the process of fusion between GAN networks and AEE operators to generate offspring, this paper adopts population entropy [64] to control the ratio of the two generations. As in [64], population entropy is used to measure the diversity of a population. Therefore, if the current population entropy is less than the historical average population entropy, increase the proportion of children generated by the GAN network; conversely, increase the proportion of children generated by the traditional operator. The calculation steps of population entropy are as follows. The similarity between an individual P_i and every other individual P_j in the population is calculated by adding up their Euclidean distances, as shown in Equation (21).

$$P_{\mathrm{i}} = \sum_{j=1}^{\mathrm{len}(P)} \left(\left| P_i - P_j \right|^2 \right) \tag{21}$$

where $len(P)$ is the population size. P_i, P_j are the individuals in the population. The expression is in two-dimensional coordinates. Then, this paper converts the similarity into probability by Gaussian distribution as follows in Equation (22):

$$P_i = exp\left\{ \frac{-P_i}{2 \cdot \sigma(P_i)} \right\} \tag{22}$$

where $\sigma(P_i)$ is the variance of individuals in the population.

After normalizing the probabilities, the entropy of the population is calculated by the following Equation (23).

$$E = -\sum_{i=1}^{\mathrm{len}(P)} (P_i * \log_2 P_i) \tag{23}$$

After obtaining the entropy of the population, it is compared with the historical average population entropy, and the proportional alpha of the hybrid operator is deflated according to the results of the comparison. The formula is shown in Equation (24).

$$alpha = \begin{cases} \max\left(\frac{alpha\_list[-1]}{scaling\_factor}, 0 \right), E < avg\_E \\ \min(alpha\_list[-1] \bullet scaling\_factor, 1), E > avg\_E \end{cases} \tag{24}$$

where $alpha\_list[-1]$ is the initial alpha value. $scaling\_factor$ is the deflation factor. avg_E is the historical average population entropy.

4. Experiments and Analysis

In this section, we firstly describe the performance indicators and the dataset and parameter configurations. We then discuss the experimental results from experiment 1 to experiment 4.

4.1. Performance Metric

4.1.1. Performance Indicators

The evaluation metrics of multi-objective evolutionary algorithms (MOEA) can be divided into three main categories: convergence metrics, distribution metrics, and comprehensive metrics. Convergence metrics, such as generational distance (GD) and inverted generational distance (IGD), are used to evaluate how well the solution set approaches the true Pareto optimal surface. Distribution metrics are used to assess the diversity and uniformity of the solution set, including the Spacing spatial evaluation method, Delta, and maximum spread (MS). The integrative metrics combine the convergence and distribution of the solution set, such as the hypervolume (HV) metric.

In this study, this paper uses six indicators, i.e., GD, IGD, Spacing, Delta, HV, and MS. Among them, GD and Spacing are used to evaluate the convergence of the algorithm, MS and Delta are used to evaluate the distributivity of the algorithm, and IGD and HV are used to evaluate the comprehensive performance of the algorithm. These six indicators are mathematically defined as follows:

1. GD: This metric measures the distance between the solution set P found by the algorithm and the reference optimal solution set P^* of the problem set. It reflects how much the solution set deviates from the true optimal bound . The larger the value, the further the deviation from the true optimal bound and the worse the convergence. The mathematical formula is as follows.

$$GD(P,P^*) = \frac{\sqrt{\sum_{y\in P} \min_{x\in P^*} \mathrm{dis}(x,y)^2}}{|P|} \tag{25}$$

   where $\mathrm{dis}(x,y)$ denotes the Euclidean distance between the y-point in the solution set P found by the algorithm and the reference optimal solution set P^*.

2. IGD: The average of the distance from each reference point to the nearest solution. The solution set with smaller IGD values is better. It can respond well to the uniformity and extensiveness of the distribution in addition to the convergence of the solution set. The smaller the IGD value, the better the diversity and convergence. The mathematical formula is as follows.

$$\mathrm{IGD}(P,P^*) = \frac{\sum_{x\in P^*} \min_{y\in P} \mathrm{dis}(x,y)}{|P^*|} \tag{26}$$

3. Spacing: This measures the standard deviation of the minimum distance of each solution to the other solutions. The smaller the Spacing value is, the more uniform the solution set is. The mathematical formula is as follows.

$$\mathrm{Spacing}(P) = \sqrt{\frac{1}{|P|-1}\sum_{\mathrm{i}=1}^{|P|}\left(\overline{\mathrm{d}} - \mathrm{d}_i\right)^2} \tag{27}$$

   where d_i denotes the minimum distance from the i-th solution of the solution set P sought by the algorithm to the other solutions, and $\overline{\mathrm{d}}$ denotes the average distance of d_i.

4. Delta: A measure of the breadth of the solution set obtained, i.e., the diversity of the solution set. The smaller the Delta value, the more diverse the solutions in the Pareto frontier set. Its mathematical formula is as follows.

$$\Delta = \frac{d_f + d_l + \sum_{i=1}^{N-1}|d_i - \bar{d}|}{d_f + d_l + (N-1)\bar{d}} \tag{28}$$

   The parameters d_f and d_l are the Euclidean distances between the extreme solutions and the boundary solutions of the obtained non-dominated set.

5. HV: The volume of the region in the target space enclosed by the set of non-dominated solutions and reference points obtained by the algorithm. HV is a comprehensive metric to assess the convergence and diversity of the approximate solution set. The larger the HV value, the better the comprehensive performance of the algorithm. The mathematical formula is as follows.

$$HV = \delta\left(\bigcup_{i=1}^{|S|} v_i\right) \tag{29}$$

where δ denotes the Lebesgue measure, which is used to measure the volume. $\mid S \mid$ denotes the number of non-dominated solution sets. v_i denotes the super volume formed by the reference point and the i-th solution in the solution set.

6. MS: This measures the maximum distance between the solutions in all solution sets. Specifically, it calculates the distance between the solutions in all solution sets and returns the maximum distance as the index value. This metric can help assess the search capability and diversity of the algorithm, as the algorithm is able to search for more solutions, and the distance between these solutions should be greater. Its mathematical formula is as follows.

$$\mathrm{MS}(P) = \sqrt{\sum_{m=1}^{M} \max(x_m - y_m)^2}, (x, y \in P) \tag{30}$$

where M is the dimension of the dataset or the number of targets. x and y are the solutions in the solution set P.

4.1.2. Dataset

To verify the effectiveness of the algorithm on high-dimensional complex data, this project uses a financial test dataset from the OR library, which is mainly applied to portfolio optimization. The dataset contains five files collecting financial indices from Hangsheng (Hong Kong), DAX (Germany), FSET (UK), S&P (US), and Nikkei (Japan). These five datasets have dimensions (i.e., number of targets or assets) of 31, 85, 89, 98, and 225, respectively, and the datasets include low-, medium-, and high-dimensional data. Our algorithm has to reweight assets based on information from the dataset to find portfolios with high returns or low risk as early as possible. The dataset can help us validate the algorithm's effectiveness on complex and high-dimensional data, as it requires handling information and relationships among up to 225 assets. The dataset includes the number of assets, the average return and the variance of returns for each asset, and the correlation between the assets. A summary of the datasets is presented in Table 1.

Table 1. A summary of the datasets used in experiments.

Dataset	Region	Dimensions
Hangsheng	Hong Kong	31
DAX100	Germany	85
FTSE100	U.K.	89
S&P100	U.S.	98
Nikkei	Japan	225

4.1.3. Comparison Algorithm

This experiment compares five multi-objective optimization algorithms (i.e., NSGA-II, MOEA/D-AEE, MOEA/D-GA, MOEA/D-D-DEM, and MOEA/D-DE [65]). NSGA-II uses non-dominance ranking and crowdedness distance to maintain the diversity of populations. MOEA/D-D-DE and MOEA/D-DEM use decomposition methods to transform the multi-objective problem into multiple single-objective sub-problems and solve these sub-problems by differential evolutionary algorithms. MOEA/D-GA uses genetic algorithms to solve multi-objective problems. All five algorithms use cross-variance for progeny propagation, and all use a single operator.

4.2. Parameter Configurations

All experiments were repeated 51 times on all five datasets. The population size was set to 100 and the neighbor size is 20. The number of offspring generated per algorithm iteration is 1500, and the upper limit of individual neighbor updates in each iteration is 2. The parameters of the AEE operator in the hybrid operator were set with reference to the literature [13], and the parameter σ used to determine the origin of the parents was set to 0.9. The initial value of alpha is 0.15. The scaling factor scaling_factor was set to 1.1.

For the parameters of the GAN network model, the learning rates of the discriminator and generator are 0.0001 and 0.0004, respectively, and the total number of iterations was set to 200 with reference to the settings in the literature [12]. This paper chose to train the GAN network every 100 generations, generating 100 individuals each time, and then perform a fast non-dominated sorting to select the top 20 individuals for replacing the parent generation. Experimental results were obtained from the last generation of the population.

4.3. Experimental Results and Analysis

4.3.1. Experiment: Comparison Analysis of the APG-SMOEA Algorithm and Other Algorithms

To better evaluate the effectiveness of the algorithm, the APG-SMOEA algorithm is compared with other multi-objective optimization algorithms. Figure 2 display the IGD variation of different algorithms on the Nikkei dataset and the populations' final distribution. Our proposed algorithm shows faster convergence in the early stage and better final results, as depicted in the figure. And the final population distribution fits the Pareto front perfectly.

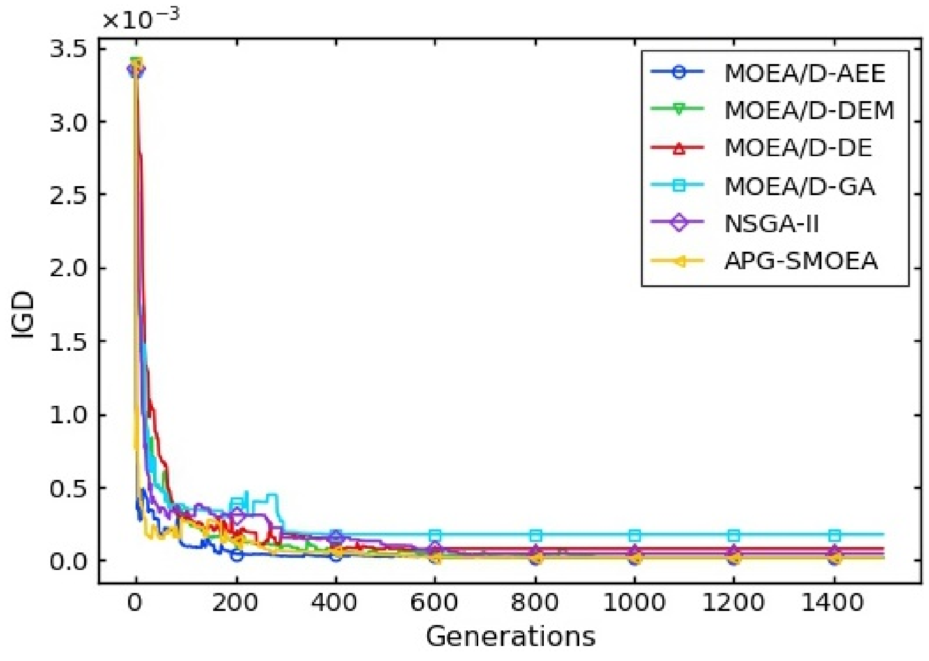

Figure 2. Changes in IGD indicators on Nikkei.

Figure 3 shows that, in the late iteration, the APG-SMOEA algorithm does not stop exploring the solution set even during the convergence to the Pareto front, and the obtained solution set fits the distribution of the Pareto front perfectly. It shows that the adaptive control strategy of population entropy is effective and effectively avoids the local optimum.

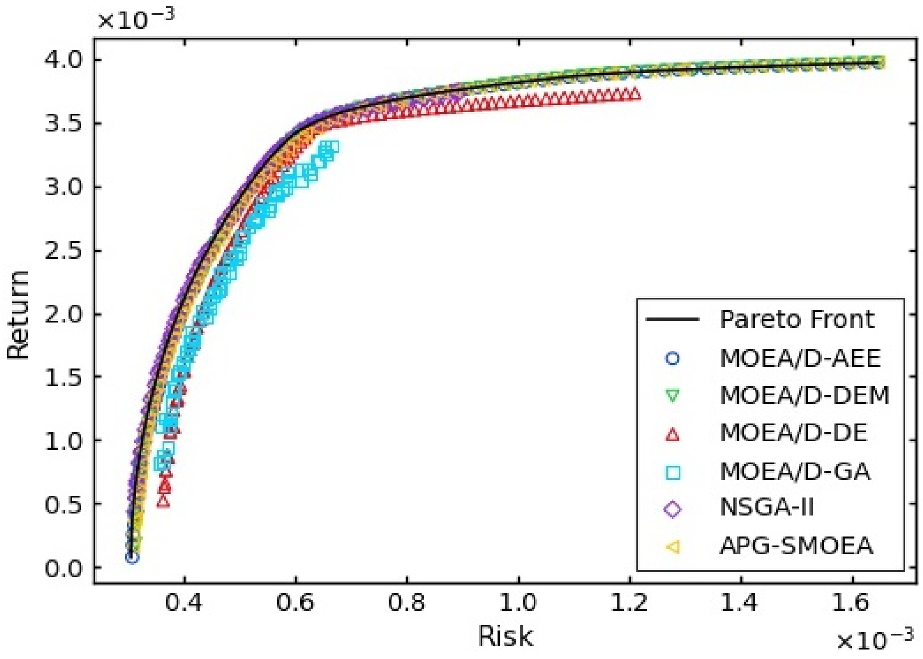

Figure 3. Final distributions of populations on Nikkei.

Figures 4 and 5 show the distribution of the population iterative process for generations 1, 2, 3, 5, 100, 150, 200, and 300. The APG-SMOEA algorithm allows a broader exploration of the space in the early iterations, with a greater diversity of populations and a wider distribution. The algorithm's search capability improves with the GAN network. In particular, training using the best samples from the population can provide a richer and more diverse solution for the population.

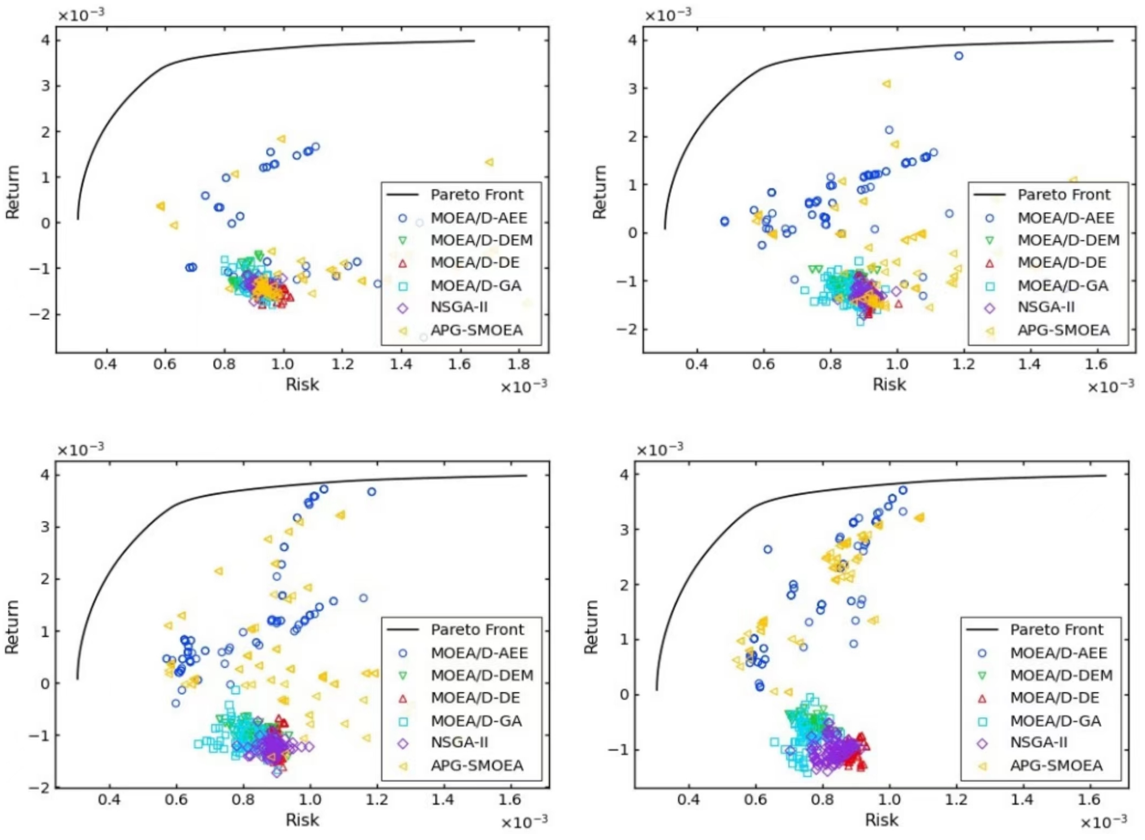

Figure 4. Distributions of populations in the 1-, 2-, 3-, and 5-generation (Nikkei).

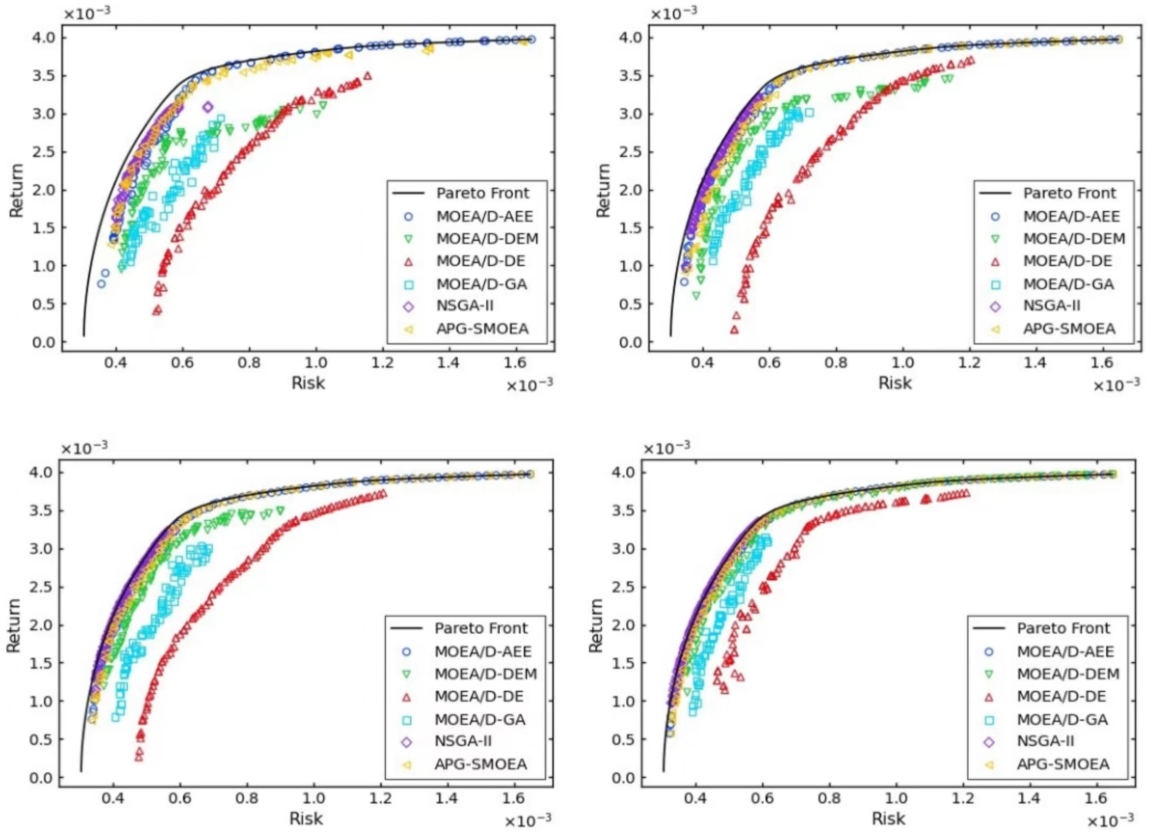

Figure 5. Distributions of populations in the 100-, 150-, 200-, and 300-generation (Nikkei).

Figure 6 shows the update frequency of the different algorithms during the iterations. The addition of the GAN network results in a decrease in the update frequency of the children; however, it still outperforms most of the other algorithms. Individuals with long trajectories (meaning those located near the ends of the Pareto front) can still be partially updated in the early iterations.

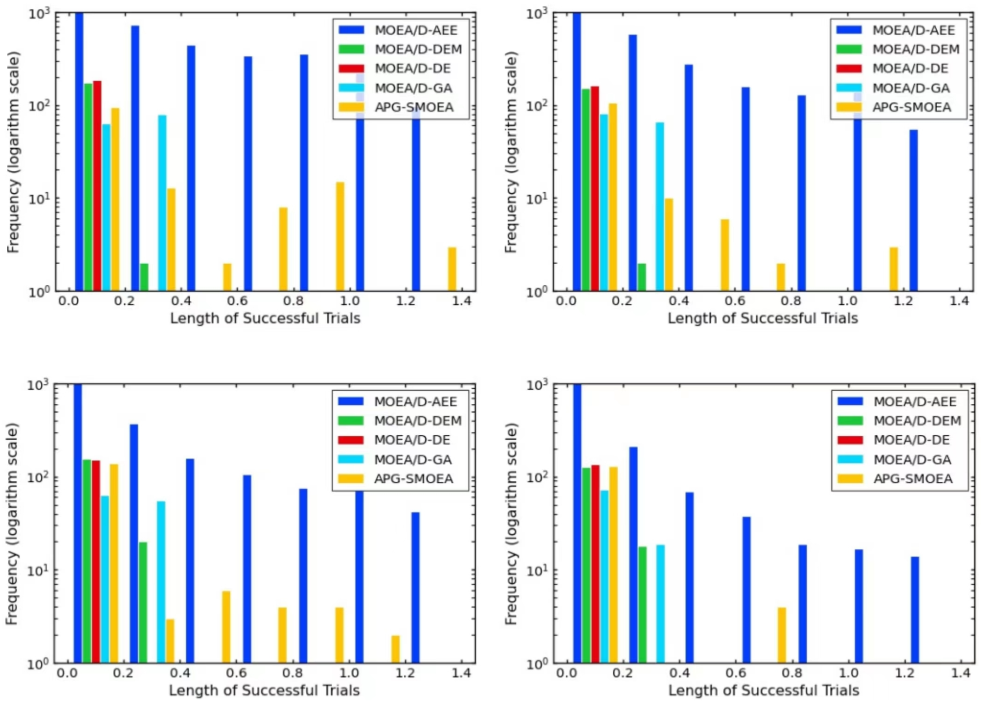

Figure 6. Frequency of successful renewal of 1-, 2-, 3-, and 5-generation offspring (Nikkei).

Tables A1–A3 (in Appendix B.1) compare the performance metrics of the APG-SMOEA algorithm with other algorithms on three datasets. Table A1 shows the comparative experimental results for the FTSE dataset, Table A2 shows the comparative experimental results for the S&P dataset, and Table A3 shows the comparative experimental results for the Nikkei dataset. It can clearly be seen that the APG-SMOEA algorithm far surpasses other algorithms in two metrics, MaxSpread and HV. It even appears exponentially higher. The APG-SMOEA algorithm has improved the performance of the GD, IGD, Delta, and Spacing metrics on the high-dimensional dataset, Nikkei, and still has significant improvement in MaxSpread and HV metrics as well.

4.3.2. Experiment II: Ablation Experiments

In this paper, ablation experiments were completed in a rigorous and comprehensive manner. The effects of offspring generated by pure GAN networks are clearly analyzed and compared with those generated by hybrid operators. This paper obtains the population distributions of both algorithms at generations 2, 3, 5, and 10 during the pre-iterative stage, as depicted in Figure 7. Figure 8 shows the population distributions of the two algorithms at the late stage of the iteration, specifically at the 100th, 150th, 200th, and 300th generations. APG-SMOEA denotes hybrid operator-generated progeny, while APG-SMOEA-15 denotes pure GAN network-generated progeny.

The experimental results show that model collapse occurs in the APG-SMOEA-15 algorithm. In the initial stage of iteration, both algorithms tend to concentrate their population distribution towards low-risk and low-return regions, with limited ability to explore the solution space. As the number of population iterations increased, APG-SMOEA successfully converged with the Pareto front and fitted the best distribution, while APG-SMOEA-15 did not converge with the Pareto front. The initial populations were generated randomly, resulting in lower quality of the non-dominated solutions obtained by fast non-dominated sorting and smaller differences between the real and generated data, which eventually led to deviations from the Pareto frontier.

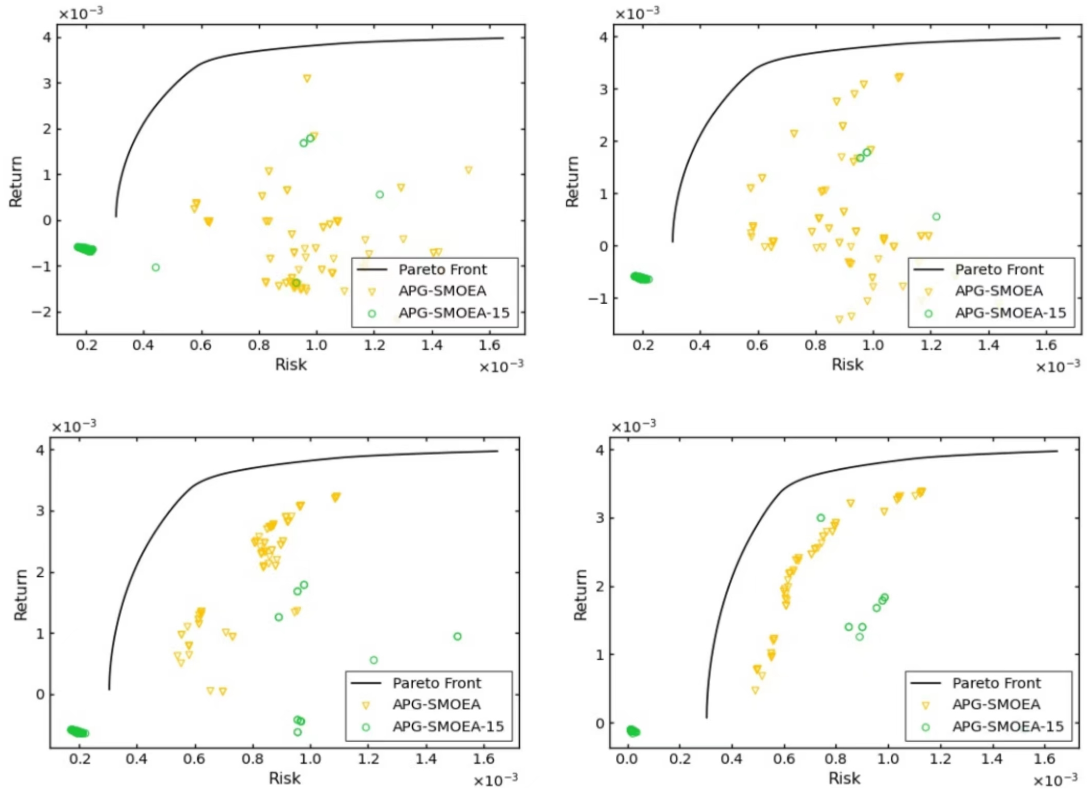

Figure 7. Distributions of populations in the 2-, 3-, 5-, and 10-generation (Nikkei).

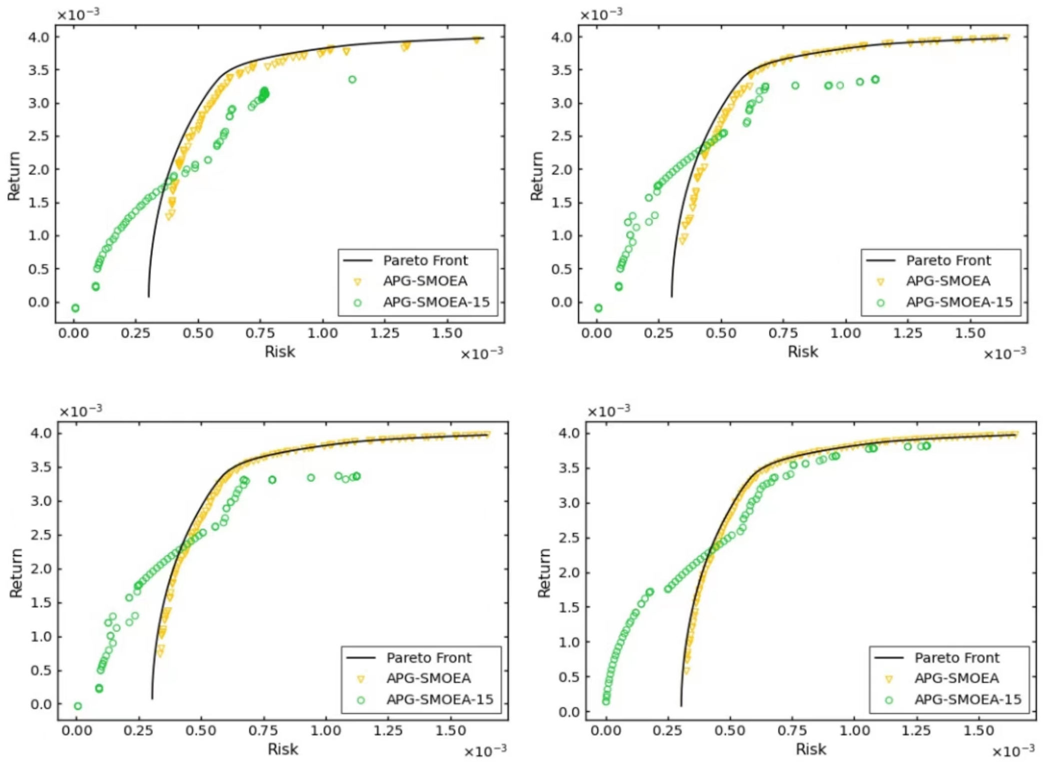

Figure 8. Distributions of populations in the 100-, 150-, 200-, and 300-generation (Nikkei).

In summary, the hybrid operator-based APG-SMOEA algorithm outperforms the pure GAN network-based algorithm in solving the multi-objective problem.

4.3.3. Experiment III: Comparative Analysis of Different Parameters of APG-SMOEA Algorithms

To obtain the optimal parameters for the APG-SMOEA algorithm, this paper uses scientific experimental methods for parameter optimization. Table 2 provides a detailed overview of the configuration details of different parameters. This section first compares four different parameter configurations of the APG-SMOEA algorithm: APG-SMOEA-1, APG-SMOEA-2, APG-SMOEA-3, and APG-SMOEA-4, to determine the optimal configuration parameters for the GAN network generation probability. This section selects experimental results from the Nikkei and Hangsheng datasets, as shown in Tables A4 and A5 (in Appendix B.2). Through analysis, it is concluded that on the Nikkei dataset, APG-SMOEA-2 algorithm performs significantly better than other algorithms, while on the Hangsheng dataset, APG-SMOEA-3 performs significantly better than other algorithms. Both the APG-SMOEA-2 and APG-SMOEA-3 algorithms have a GAN network generation probability of 0.1. Therefore, the conclusion is drawn that when the generation probability of the generative adversarial network is set to 0.1, the algorithm performs optimally.

Table 2. Experimental parameters configuration.

Algorithm	**GAN Generation Probability**	**GAN Network Training Period/Algebra**	**The Number of Individuals Used to Replace the Parent**
APG-SMOEA-1	0.3	100	First 50 individuals
APG-SMOEA-2	0.1	100	First 50 individuals
APG-SMOEA-3	0.1	20	First 10 individuals
APG-SMOEA-4	0.1–0.9 uniform change	100	First 50 individuals
APG-SMOEA-5	0.1	100	First 20 individuals
APG-SMOEA-6	0.1	50	First 50 individuals
APG-SMOEA-7	0.05	100	First 50 individuals

To determine the optimal settings for two parameters of the generative adversarial network, namely, the GAN network training period/algebra and the number of individuals used to replace the parent, we further designed three algorithms with different parameter settings: APG-SMOEA-5, APG-SMOEA-6, and APG-SMOEA-7. We conducted comparative experiments and the results are shown in Tables A6 and A7 (in Appendix B.2). This section still selects the high-dimensional data of the Nikkei index and the Hangsheng index as a comparison to analyze the influence of these two parameters on the training period of the GAN network and the number of individuals used to replace the parent when the GAN network's generation probability is 0.1. Through comparison, it is found that on the Hangsheng dataset, the APG-SMOEA-2 algorithm performs best, while on the Nikkei dataset, the APG-SMOEA-5 performs best. Especially, as a high-dimensional dataset, Nikkei has achieved all optimal results in various performance indicators with APG-SMOEA-5. Therefore, we draw the conclusion that the optimal parameters for the APG-SMOEA algorithm are 0.1 for the GAN network's generation probability, 100 for the GAN network's training period, and 20 for the number of individuals used to replace the parent. In this paper, we will use these optimal parameters as the parameters of the APG-SMOEA algorithm.

4.3.4. Experiment IV: APG-SMOEA Algorithm Improvement Exploration

With the experimental results in Section 4.3.3, this paper derives the optimal parameters for generating offspring by GANs. However, fixed parameters may not be a better choice in general. We further explore the generative probabilities of the GAN network. And the specific parameter configurations are shown in Table 3. Uniform variation refers to a gradual increase in parameters over generations, while interval variation means that different generations have fixed probability values. The experimental results in Table A8 (in Appendix B.3) do not show the expected improvements, indicating that a simple algebraic variation of GAN generation probabilities is not the optimal choice.

Table 3. Experimental parameters configuration by algebraic variation.

Algorithm	GAN Generation Probability	GAN Network Training Period/Algebra	The Number of Individuals Used to Replace the Parent
APG-SMOEA-8	0.1–0.9 uniform change	100	First 20 individuals
APG-SMOEA-9	0.1–0.9 uniform change	50	First 50 individuals
APG-SMOEA-10	0.1–0.9 interval change	100	First 50 individuals
APG-SMOEA-11	0.1–0.3 interval change	100	First 50 individuals

Based on the above failed experiences, this paper tries to use adaptive control methods by three different implementations: the first one based on Gaussian distribution changes [66], the second one based on the proportion of successfully updated generated offspring, and the last one based on population entropy strategies [67].

Regarding the Gaussian distribution variation methods (APG-SMOEA-12), when the GAN generation probability is between 0.1 and 0.3, the algorithm can obtain better performance in this interval. Therefore, this paper uses a Gaussian distribution with a mean value of 0.15 to generate 1000 random numbers and filter out values greater than 1 or less than 0, from which we randomly select probability values as the generation probability of the GAN network.

With regard to the proportion of successfully updated generated offspring methods (APG-SMOEA-13), this paper records the number of individuals that successfully update their parents in each generation of GAN-generated offspring, and calculate their proportion of the total number of individuals. If this ratio is greater than 0.35, it indicates that the GAN network is well trained and the generation probability of the GAN network needs to be increased.

The population entropy control methods (APG-SMOEA-14) can measure the diversity of populations. If the overall entropy of a generation is higher than the historical average, it indicates that the diversity of the population is better. In this case, the GAN network's generation probability needs to be increased to provide more diverse training samples.

The experimental results show that APG-SMOEA-14 is the best one among the three algorithms, for example in Tables A9 and A10 in the FTSE and DAX dataset (in Appendix B.3). In view of the situation that the performance of the APG-SMOEA is not particularly satisfactory, inspired by [68], we also propose a learning pool approach. Specifically, the learning pool serves as a collection that stores the best-performing individuals within the population. By utilizing these excellent individuals as training samples, we aim to increase the diversity and quality of samples, thereby improving the generative performance of the network. However, the final experimental results were not satisfactory, and we will make further relevant attempts.

In Section 4.3.1, we compared the APG-SMOEA algorithm with other MOEA algorithms by experiments. The results show that the APG-SMOEA algorithm exhibits excellent performance on high-dimensional data and can significantly improve the exploration ability of the solution space. As a result, the improvements in both MaxSpread and HV evaluation metrics are large, with exponential improvements observed on some datasets.

In Section 4.3.2, we performed ablation experiments to verify the necessity of the hybrid operator. The experimental results show that the generation of offspring from pure GAN networks leads to the problem of pattern collapse.

In Sections 4.3.3 and 4.3.4, this paper describes the work conducted in the process of improving the APG-SMOEA algorithm. Section 4.3.3 is an a priori experiment conducted to determine the hyperparameters of the algorithm. In Section 4.3.4, we proposed three implementation approaches: Gaussian distribution-based variations, proportion-based adjustments using the success rate of generated offspring updates, and adaptive variations based on population entropy. The effectiveness of the adaptive parameters is verified by experiments. And we decide to use population entropy.

5. Conclusions

This paper proposes a synergistic MOEA with adversarial population generation, namely, APG-SMOEA, to address complex data issues. The proposed algorithm utilizes the complementary strengths of MOEAs and GANs to optimize MOPs simultaneously while effectively exploring the solution space in high-dimensional and non-linear problem domains. In particular, this paper introduces an adaptive population entropy strategy to balance the efficiency and quality of offspring selection.

The experimental results show that the APG-SMOEA algorithm is more effective on complex data on both low and high dimensions, and is superior to the most advanced MOEA algorithms on various benchmark problems. The proposed algorithm not only showed superior diversity performance but also exhibited robustness and adaptability to different complexity problems. For future work, several potential research directions can be identified. Firstly, we can study the potential synergies between MOEAs, GANs, and other optimization techniques to develop novel hybrid algorithms with superior performance. Secondly, distributed computing can be employed to speed up the algorithm's execution, so that it can be applied to more demanding optimization tasks when dealing with complex data. Thirdly, we can enhance the interpretability of APG-SMOEA by integrating various techniques that help to understand the components of the algorithm, their interactions, and the decision-making process. Lastly, we can explore various methods, such as the neurodynamic approach [23] and reinforcement learning algorithm [69], to improve asset allocation decisions by accounting for multiple conflicting objectives, ultimately enhancing portfolio performance and addressing the complexities associated with financial markets.

Author Contributions: Methodology, W.Q.; Software, H.C.; Validation, R.X. and M.L.; Formal analysis, H.X. and M.Y.; Investigation, L.Y.; Resources, L.Y. and M.L.; Writing—original draft, W.Q., H.C. and Y.L.; Writing—review & editing, H.X., Y.L. and R.X. All authors have read and agreed to the published version of the manuscript.

Funding: This paper is supported by the 2021 Fujian Foreign Cooperation Project (No. 2021I0001): Research on Human Behavior Recognition Based on RFID and Deep Learning; the Horizontal project (Co-construction platform), 2023 Project of Xiamen University: Joint Laboratory of Public Safety and Artificial Intelligence (20233160C003); the State Key Laboratory of Process Automation in Mining & Metallurgy, Beijing Key Laboratory of Process Automation in Mining & Metallurgy (No. BGRIMM-KZSKL-2022-14): Research and application of mine operator positioning based on RFID and deep learning; the National Key R&D Program of China-Sub-project of Major Natural Disaster Monitoring, Early Warning and Prevention (No. 2020YFC1522604): Research on key technologies of comprehensive information application platform for cultural relic safety based on big data technology; the Scientific Research Project of Putian Science and Technology Bureau, 2021ZP07; the National Natural Science Foundation of China, 62103209; and the Natural Science Foundation of Fujian Province, 2020J05213.

Data Availability Statement: http://people.brunel.ac.uk/~mastjjb/jeb/orlib/portinfo.html (accessed on 1 January 2024).

Conflicts of Interest: The authors declare no conflict of interest.

Appendix A. The Algorithms

Appendix A.1. APG-SMOEA Algorithm Framework

Algorithm A1 APG-SMOEA algorithm framework

```
Input: $T$ is the maximum number of iterations, $n_r$ is the maximum number of updates of
    the offspring;
Initialize: $T = 1500$, $n_r = 2$;
 1: Read the dataset parameters
 2: Generate initial population
 3: Calculate initial parameters
 4: Build GAN networks
 5: while $t < T$ do
 6:   for $x_i$ in population do
 7:     Determine the mating pool b of offspring produced by the variant operator
 8:     if the $Rn > \alpha$ then
 9:       AEE operator generates offspring (Equation (15))
10:     else
11:       GAN networks generate offspring (Algorithm A2, in Appendix A.2)
12:     end if
13:     Repair offspring
14:     Calculate information of offspring (e.g., return, risk, reference points)
15:     Calculate and update initial parameters
16:     for $b_i$ in neighbor of $b$ do
17:       if the offspring M, V is better than the parent $b_i$ then
18:         Update populations,replace $b_i$ with $y$
19:         $update\_count + 1$
20:       end if
21:       if $update\_count >= n_r$ then
22:         Break;
23:       end if
24:     end for
25:     Update $\alpha$ (Algorithm A3, in Appendix A.3)
26:   end for
27: end while
```

Appendix A.2. GAN Networks Generate Offspring

Algorithm A2 GAN networks generate offspring

```
Input: $max\_num$ is the upper limit of the child selection, period is the training period of
    GAN network, $t$ is the population iteration times.
Initialize: $max\_num = 20$, $period = 100$.
 1: if $k = max\_num$ or $k = 0$ then
 2:     Fast non-dominated sorting of populations
 3:     Record the top ten individuals in the population
 4:     Mark the top 10 individuals as excellent individuals
 5:     Generate a mating pool containing the best individuals as real data for training
 6:     Generate input data containing both true data (top ten best individuals) and false
        data (dominated solutions)
 7: end if
 8: if $t \% period = 0$ or $t = 0$ then
 9:     Train GAN networks (Equation (16) )
10:     Generate offspring using GAN (Equation (20))
11:     Perform a fast non-dominated sort on the generated offspring, and use the first
        twenty individuals to replace the parent
12: end if
```

Appendix A.3. Update Alpha

Algorithm A3 Update alpha

```
Input: $E$ is the entropy of the population, $\alpha$ is the ratio of mixed operators, $scaling\_factor$
    is the scaling factor of $\alpha$.
Initialize: $\alpha = 0.15$, $scaling\_factor = 1.1$.
 1: Initialize each individual with probability $P$ to 0
 2: Calculate the Euclidean distance between individual $P_i$ and other individuals $P_j$ in
    the summation population to obtain the similarity of individuals in the population
    (Equation (21))
 3: Converting similarity to probability via Gaussian distribution
 4: Normalization of probabilities (Equation (22))
 5: Calculating the entropy of a population by probability (Equation (23))
 6: if $E < avg\_E$ then
 7:     $\alpha$ = max(alpha_list[-1] /$scaling\_factor$, 0) (Equation (24))
 8: else
 9:     $\alpha$ = min(alpha_list[-1] * $scaling\_factor$, 1) (Equation (24))
10: end if
11: Regularize the $\alpha$ so that the $\alpha$ regresses towards the initial $\alpha$
```

Appendix B. The Tables

Appendix B.1. Tables in Experiment 1

Table A1. Comparative experimental results for the FTSE dataset.

Metric		MOEA/D-AEE	MOEA/D-DEM	MOEA/D-DE	MOEA/D-GA	NSGA-II	APG-SMOEA
	Best	5.27×10^{-6}	6.32×10^{-6}	7.01×10^{-6}	**2.84×10^{-6}**	7.38×10^{-6}	4.58×10^{-3}
GD	Median	7.05×10^{-6}	9.58×10^{-6}	1.83×10^{-5}	**5.05×10^{-6}**	9.25×10^{-6}	7.98×10^{-1}
	Std.	**7.28×10^{-7}**	2.41×10^{-6}	1.55×10^{-4}	1.60×10^{-6}	9.50×10^{-7}	2.61×10^{-1}
	Best	1.71×10^{-5}	1.38×10^{-5}	**9.87×10^{-6}**	1.14×10^{-5}	2.39×10^{-5}	1.53×10^{-4}
Spacing	Median	2.07×10^{-5}	2.01×10^{-5}	**1.72×10^{-5}**	1.97×10^{-5}	2.99×10^{-5}	1.12×10^{-2}
	Std.	3.76×10^{-6}	4.54×10^{-6}	3.34×10^{-6}	5.10×10^{-6}	**2.24×10^{-6}**	1.20×10^{-2}
	Best	5.96×10^{-3}	5.79×10^{-3}	5.74×10^{-3}	5.47×10^{-3}	5.67×10^{-3}	**2.27×10^{0}**
Max Spread	Median	5.56×10^{-3}	5.40×10^{-3}	5.15×10^{-3}	4.91×10^{-3}	5.45×10^{-3}	**1.96×10^{0}**
	Std.	**1.59×10^{-4}**	1.93×10^{-4}	5.08×10^{-4}	4.19×10^{-4}	1.69×10^{-4}	6.28×10^{-1}
	Best	**4.01×10^{-1}**	4.25×10^{-1}	4.21×10^{-1}	4.27×10^{-1}	5.47×10^{-1}	6.60×10^{-1}
Delta	Median	**4.33×10^{-1}**	4.72×10^{-1}	4.51×10^{-1}	5.05×10^{-1}	6.06×10^{-1}	8.78×10^{-1}
	Std.	**2.02×10^{-2}**	3.06×10^{-2}	6.30×10^{-2}	6.88×10^{-2}	3.34×10^{-2}	1.16×10^{-1}
	Best	**2.30×10^{-5}**	2.90×10^{-5}	4.26×10^{-5}	4.07×10^{-5}	3.22×10^{-5}	7.59×10^{-5}
IGD	Median	**3.71×10^{-5}**	5.31×10^{-5}	9.09×10^{-5}	8.76×10^{-5}	4.74×10^{-5}	1.08×10^{-3}
	Std.	**1.14×10^{-5}**	2.16×10^{-5}	1.47×10^{-4}	4.33×10^{-5}	1.38×10^{-5}	3.65×10^{-4}
	Best	1.96×10^{-2}	1.96×10^{-2}	1.96×10^{-2}	1.96×10^{-2}	1.96×10^{-2}	**3.93×10^{-1}**
HV	Median	1.96×10^{-2}	1.96×10^{-2}	1.96×10^{-2}	1.92×10^{-2}	1.96×10^{-2}	**3.67×10^{-1}**
	Std.	**1.08×10^{-8}**	2.50×10^{-6}	1.43×10^{-3}	8.49×10^{-4}	1.45×10^{-4}	8.88×10^{-2}

Table A2. Comparative experimental results for the S&P dataset.

Metric		MOEA/D-AEE	MOEA/D-DEM	MOEA/D-DE	MOEA/D-GA	NSGA-II	APG-SMOEA
	Best	9.36×10^{-6}	1.13×10^{-5}	1.27×10^{-5}	**3.20×10^{-6}**	1.01×10^{-5}	7.92×10^{-3}
GD	Median	1.21×10^{-5}	1.75×10^{-5}	3.92×10^{-5}	**5.02×10^{-6}**	1.29×10^{-5}	6.76×10^{-1}
	Std.	1.55×10^{-6}	3.45×10^{-6}	7.64×10^{-5}	1.86×10^{-6}	**1.42×10^{-6}**	1.98×10^{-1}
	Best	2.25×10^{-5}	2.01×10^{-5}	**1.72×10^{-5}**	1.77×10^{-5}	2.51×10^{-5}	2.75×10^{-4}
Spacing	Median	2.66×10^{-5}	2.76×10^{-5}	**2.31×10^{-5}**	2.35×10^{-5}	3.56×10^{-5}	9.55×10^{-3}
	Std.	5.45×10^{-6}	5.26×10^{-6}	4.78×10^{-6}	5.06×10^{-6}	**3.71×10^{-6}**	8.21×10^{-3}
	Best	7.70×10^{-3}	7.86×10^{-3}	7.60×10^{-3}	6.87×10^{-3}	7.29×10^{-3}	**1.99×10^{0}**
Max Spread	Median	7.41×10^{-3}	7.55×10^{-3}	7.23×10^{-3}	5.82×10^{-3}	6.57×10^{-3}	**1.62×10^{0}**
	Std.	**1.30×10^{-4}**	1.61×10^{-4}	2.53×10^{-4}	4.35×10^{-4}	3.52×10^{-4}	4.82×10^{-1}
	Best	**3.29×10^{-1}**	3.30×10^{-1}	**3.29×10^{-1}**	4.22×10^{-1}	5.37×10^{-1}	6.61×10^{-1}
Delta	Median	**3.53×10^{-1}**	3.74×10^{-1}	3.66×10^{-1}	5.57×10^{-1}	6.48×10^{-1}	8.59×10^{-1}
	Std.	**1.77×10^{-2}**	3.37×10^{-2}	2.41×10^{-2}	3.88×10^{-2}	3.69×10^{-2}	1.34×10^{-1}
	Best	**3.21×10^{-5}**	3.30×10^{-5}	3.64×10^{-5}	4.88×10^{-5}	4.08×10^{-5}	1.73×10^{-4}
IGD	Median	**4.15×10^{-5}**	4.35×10^{-5}	7.12×10^{-5}	1.36×10^{-4}	7.08×10^{-5}	1.28×10^{-3}
	Std.	5.77×10^{-6}	7.65×10^{-6}	4.16×10^{-5}	4.13×10^{-5}	**2.21×10^{-5}**	3.59×10^{-4}
	Best	1.83×10^{-2}	1.83×10^{-2}	1.83×10^{-2}	1.82×10^{-2}	1.82×10^{-2}	**3.90×10^{-1}**
HV	Median	1.83×10^{-2}	1.83×10^{-2}	1.83×10^{-2}	1.69×10^{-2}	1.74×10^{-2}	**3.65×10^{-1}**
	Std.	**1.75×10^{-8}**	3.47×10^{-8}	4.10×10^{-4}	6.01×10^{-4}	4.32×10^{-4}	7.31×10^{-2}

Table A3. Comparative experimental results for the Nikkei dataset.

Metric		MOEA/D-AEE	MOEA/D-DEM	MOEA/D-DE	MOEA/D-GA	NSGA-II	APG-SMOEA
GD	Best	5.32×10^{-6}	5.10×10^{-6}	5.93×10^{-5}	9.95×10^{-6}	$\mathbf{2.54 \times 10^{-6}}$	6.13×10^{-6}
	Median	7.36×10^{-6}	8.06×10^{-6}	1.45×10^{-4}	3.32×10^{-5}	$\mathbf{4.24 \times 10^{-6}}$	8.54×10^{-6}
	Std.	$\mathbf{9.19 \times 10^{-7}}$	1.66×10^{-4}	7.09×10^{-5}	2.08×10^{-4}	2.01×10^{-6}	3.76×10^{-4}
Spacing	Best	1.42×10^{-5}	1.28×10^{-5}	$\mathbf{5.99 \times 10^{-6}}$	0.00×10^{0}	1.05×10^{-5}	1.38×10^{-5}
	Median	1.79×10^{-5}	2.08×10^{-5}	$\mathbf{1.15 \times 10^{-5}}$	1.92×10^{-5}	1.45×10^{-5}	2.03×10^{-5}
	Std.	4.87×10^{-6}	5.77×10^{-6}	4.53×10^{-6}	9.72×10^{-6}	$\mathbf{1.87 \times 10^{-6}}$	3.84×10^{-5}
Max Spread	Best	4.17×10^{-3}	4.23×10^{-3}	4.29×10^{-3}	2.63×10^{-3}	3.36×10^{-3}	$\mathbf{1.16 \times 10^{-2}}$
	Median	3.93×10^{-3}	3.94×10^{-3}	2.96×10^{-3}	2.20×10^{-3}	2.88×10^{-3}	$\mathbf{4.01 \times 10^{-3}}$
	Std.	$\mathbf{1.21 \times 10^{-4}}$	5.39×10^{-4}	5.48×10^{-4}	4.65×10^{-4}	2.54×10^{-4}	1.15×10^{-3}
Delta	Best	3.87×10^{-1}	3.99×10^{-1}	$\mathbf{3.17 \times 10^{-1}}$	8.40×10^{-1}	6.09×10^{-1}	3.99×10^{-1}
	Median	$\mathbf{4.42 \times 10^{-1}}$	4.81×10^{-1}	5.58×10^{-1}	9.34×10^{-1}	6.81×10^{-1}	4.55×10^{-1}
	Std.	$\mathbf{6.09 \times 10^{-2}}$	9.05×10^{-2}	1.00×10^{-1}	3.55×10^{-2}	2.94×10^{-2}	1.42×10^{-1}
IGD	Best	$\mathbf{1.72 \times 10^{-5}}$	1.90×10^{-5}	7.90×10^{-5}	1.77×10^{-4}	4.64×10^{-5}	1.84×10^{-5}
	Median	$\mathbf{2.47 \times 10^{-5}}$	2.73×10^{-5}	2.23×10^{-4}	2.41×10^{-4}	9.69×10^{-5}	2.75×10^{-5}
	Std.	$\mathbf{7.05 \times 10^{-6}}$	4.09×10^{-4}	6.95×10^{-5}	5.29×10^{-4}	3.67×10^{-5}	2.84×10^{-5}
HV	Best	2.06×10^{-4}	2.06×10^{-4}	2.05×10^{-4}	1.98×10^{-4}	1.99×10^{-4}	$\mathbf{3.85 \times 10^{-4}}$
	Median	2.06×10^{-4}	2.06×10^{-4}	1.98×10^{-4}	1.86×10^{-4}	1.91×10^{-4}	$\mathbf{2.06 \times 10^{-4}}$
	Std.	$\mathbf{1.06 \times 10^{-8}}$	2.12×10^{-5}	7.70×10^{-6}	2.64×10^{-5}	3.08×10^{-6}	2.69×10^{-5}

Appendix B.2. Tables in Experiment 3

Table A4. Experimental results for the Hangsheng dataset.

Metric		APG-SMOEA-1	APG-SMOEA-2	APG-SMOEA-3	APG-SMOEA-4
GD	Best	1.36×10^{-2}	$\mathbf{7.05 \times 10^{-5}}$	2.49×10^{-4}	1.53×10^{-4}
	Median	4.11×10^{-1}	4.20×10^{-1}	$\mathbf{3.96 \times 10^{-1}}$	4.11×10^{-1}
	Std.	$\mathbf{8.97 \times 10^{-2}}$	1.06×10^{-1}	1.70×10^{-1}	1.18×10^{-1}
Spacing	Best	1.12×10^{-3}	5.63×10^{-5}	$\mathbf{4.45 \times 10^{-5}}$	5.17×10^{-5}
	Median	6.81×10^{-3}	6.86×10^{-3}	5.95×10^{-3}	$\mathbf{5.66 \times 10^{-3}}$
	Std.	5.86×10^{-3}	$\mathbf{4.78 \times 10^{-3}}$	5.70×10^{-3}	6.72×10^{-3}
Max Spread	Best	$\mathbf{1.09 \times 10^{0}}$	$\mathbf{1.09 \times 10^{0}}$	$\mathbf{1.09 \times 10^{0}}$	$\mathbf{1.09 \times 10^{0}}$
	Median	1.04×10^{0}	1.05×10^{0}	$\mathbf{1.06 \times 10^{0}}$	1.05×10^{0}
	Std.	$\mathbf{1.68 \times 10^{-1}}$	2.39×10^{-1}	3.91×10^{-1}	2.85×10^{-1}
Delta	Best	7.12×10^{-1}	6.11×10^{-1}	$\mathbf{4.48 \times 10^{-1}}$	6.10×10^{-1}
	Median	9.48×10^{-1}	9.48×10^{-1}	9.49×10^{-1}	$\mathbf{9.12 \times 10^{-1}}$
	Std.	$\mathbf{1.51 \times 10^{-1}}$	1.67×10^{-1}	1.98×10^{-1}	2.05×10^{-1}
IGD	Best	1.72×10^{-4}	3.94×10^{-5}	$\mathbf{3.81 \times 10^{-5}}$	3.82×10^{-5}
	Median	6.38×10^{-4}	6.21×10^{-4}	$\mathbf{5.97 \times 10^{-4}}$	6.43×10^{-4}
	Std.	$\mathbf{7.90 \times 10^{-5}}$	1.31×10^{-4}	1.80×10^{-4}	1.40×10^{-4}
HV	Best	8.35×10^{-2}	8.43×10^{-2}	$\mathbf{8.91 \times 10^{-2}}$	8.32×10^{-2}
	Median	7.70×10^{-2}	7.66×10^{-2}	$\mathbf{8.24 \times 10^{-2}}$	7.65×10^{-2}
	Std.	$\mathbf{7.76 \times 10^{-3}}$	1.36×10^{-2}	2.38×10^{-2}	1.66×10^{-2}

Table A5. Experimental results for the Nikkei dataset.

Metric		**APG-SMOEA-1**	**APG-SMOEA-2**	**APG-SMOEA-3**	**APG-SMOEA-4**
GD	Best	9.40×10^{-6}	$\mathbf{7.15 \times 10^{-6}}$	2.31×10^{-5}	8.79×10^{-6}
	Median	3.23×10^{-5}	3.14×10^{-5}	7.81×10^{-5}	$\mathbf{2.92 \times 10^{-5}}$
	Std.	1.37×10^{-3}	$\mathbf{2.34 \times 10^{-5}}$	1.90×10^{-3}	1.02×10^{-3}
Spacing	Best	$\mathbf{1.33 \times 10^{-5}}$	1.41×10^{-5}	1.90×10^{-5}	1.49×10^{-5}
	Median	2.31×10^{-5}	$\mathbf{2.30 \times 10^{-5}}$	2.70×10^{-5}	2.33×10^{-5}
	Std.	6.86×10^{-5}	$\mathbf{6.64 \times 10^{-6}}$	5.25×10^{-5}	5.28×10^{-5}
Max Spread	Best	$\mathbf{3.37 \times 10^{-2}}$	9.25×10^{-3}	2.43×10^{-2}	1.88×10^{-2}
	Median	4.24×10^{-3}	$\mathbf{4.27 \times 10^{-3}}$	$\mathbf{4.27 \times 10^{-3}}$	4.26×10^{-3}
	Std.	4.21×10^{-3}	$\mathbf{7.25 \times 10^{-4}}$	4.44×10^{-3}	2.58×10^{-3}
Delta	Best	$\mathbf{3.87 \times 10^{-1}}$	3.93×10^{-1}	4.35×10^{-1}	4.13×10^{-1}
	Median	5.11×10^{-1}	$\mathbf{5.07 \times 10^{-1}}$	6.10×10^{-1}	5.26×10^{-1}
	Std.	1.35×10^{-1}	$\mathbf{1.15 \times 10^{-1}}$	2.11×10^{-1}	1.50×10^{-1}
IGD	Best	2.08×10^{-5}	$\mathbf{2.02 \times 10^{-5}}$	4.69×10^{-5}	2.11×10^{-5}
	Median	3.18×10^{-5}	$\mathbf{2.58 \times 10^{-5}}$	1.18×10^{-4}	3.00×10^{-5}
	Std.	4.79×10^{-5}	$\mathbf{2.80 \times 10^{-5}}$	3.70×10^{-5}	2.90×10^{-5}
HV	Best	4.22×10^{-4}	2.44×10^{-4}	$\mathbf{4.25 \times 10^{-4}}$	4.04×10^{-4}
	Median	$\mathbf{2.07 \times 10^{-4}}$	$\mathbf{2.07 \times 10^{-4}}$	$\mathbf{2.07 \times 10^{-4}}$	2.06×10^{-4}
	Std.	3.51×10^{-5}	$\mathbf{5.27 \times 10^{-6}}$	7.05×10^{-5}	4.58×10^{-5}

Table A6. Experimental results for the Hangsheng dataset where GAN generation probability is 0.1.

Metric		**APG-SMOEA-2**	**APG-SMOEA-5**	**APG-SMOEA-6**	**APG-SMOEA-7**
GD	Best	$\mathbf{7.05 \times 10^{-5}}$	1.59×10^{-4}	1.95×10^{-4}	1.18×10^{-4}
	Median	4.20×10^{-1}	4.22×10^{-1}	$\mathbf{3.87 \times 10^{-1}}$	3.91×10^{-1}
	Std.	$\mathbf{1.06 \times 10^{-1}}$	1.24×10^{-1}	1.35×10^{-1}	1.39×10^{-1}
Spacing	Best	$\mathbf{5.63 \times 10^{-5}}$	5.81×10^{-5}	1.21×10^{-4}	6.66×10^{-5}
	Median	6.86×10^{-3}	6.26×10^{-3}	$\mathbf{5.87 \times 10^{-3}}$	7.12×10^{-3}
	Std.	$\mathbf{4.78 \times 10^{-3}}$	5.52×10^{-3}	5.09×10^{-3}	6.87×10^{-3}
Max Spread	Best	$\mathbf{1.09 \times 10^{0}}$	$\mathbf{1.09 \times 10^{0}}$	$\mathbf{1.09 \times 10^{0}}$	$\mathbf{1.09 \times 10^{0}}$
	Median	$\mathbf{1.05 \times 10^{0}}$	$\mathbf{1.05 \times 10^{0}}$	1.02×10^{0}	9.93×10^{-1}
	Std.	$\mathbf{2.39 \times 10^{-1}}$	2.76×10^{-1}	2.98×10^{-1}	3.09×10^{-1}
Delta	Best	6.11×10^{-1}	6.41×10^{-1}	7.58×10^{-1}	$\mathbf{5.78 \times 10^{-1}}$
	Median	9.48×10^{-1}	$\mathbf{9.06 \times 10^{-1}}$	9.24×10^{-1}	9.88×10^{-1}
	Std.	1.67×10^{-1}	1.72×10^{-1}	$\mathbf{1.40 \times 10^{-1}}$	2.02×10^{-1}
IGD	Best	3.94×10^{-5}	4.11×10^{-5}	9.90×10^{-5}	$\mathbf{3.58 \times 10^{-5}}$
	Median	6.21×10^{-4}	6.43×10^{-4}	6.01×10^{-4}	$\mathbf{5.98 \times 10^{-4}}$
	Std.	$\mathbf{1.31 \times 10^{-4}}$	1.36×10^{-4}	1.32×10^{-4}	1.63×10^{-4}
HV	Best	8.43×10^{-2}	8.26×10^{-2}	$\mathbf{8.61 \times 10^{-2}}$	8.26×10^{-2}
	Median	7.66×10^{-2}	7.65×10^{-2}	$\mathbf{7.83 \times 10^{-2}}$	7.61×10^{-2}
	Std.	1.36×10^{-2}	1.47×10^{-2}	1.59×10^{-2}	$\mathbf{1.82 \times 10^{-2}}$

Table A7. Experimental results for the Nikkei dataset where GAN generation probability is 0.1.

Metric		APG-SMOEA-2	APG-SMOEA-5	APG-SMOEA-6	APG-SMOEA-7
GD	Best	$\mathbf{7.15 \times 10^{-6}}$	$\mathbf{7.15 \times 10^{-6}}$	1.80×10^{-5}	9.92×10^{-6}
	Median	3.14×10^{-5}	$\mathbf{2.58 \times 10^{-5}}$	4.04×10^{-5}	3.30×10^{-5}
	Std.	2.34×10^{-5}	8.61×10^{-4}	5.25×10^{-3}	1.98×10^{-3}
Spacing	Best	$\mathbf{1.41 \times 10^{-5}}$	1.57×10^{-5}	1.73×10^{-5}	1.63×10^{-5}
	Median	2.30×10^{-5}	$\mathbf{2.22 \times 10^{-5}}$	2.34×10^{-5}	2.27×10^{-5}
	Std.	$\mathbf{6.64 \times 10^{-6}}$	2.87×10^{-5}	2.33×10^{-4}	3.43×10^{-5}
Max Spread	Best	9.25×10^{-3}	2.23×10^{-2}	$\mathbf{1.55 \times 10^{-1}}$	3.89×10^{-2}
	Median	4.27×10^{-3}	$\mathbf{4.28 \times 10^{-3}}$	4.22×10^{-3}	4.26×10^{-3}
	Std.	7.25×10^{-4}	2.56×10^{-3}	$\mathbf{2.11 \times 10^{-2}}$	6.21×10^{-3}
Delta	Best	3.93×10^{-1}	$\mathbf{3.87 \times 10^{-1}}$	4.24×10^{-1}	4.02×10^{-1}
	Median	5.07×10^{-1}	$\mathbf{4.99 \times 10^{-1}}$	5.14×10^{-1}	5.05×10^{-1}
	Std.	1.15×10^{-1}	$\mathbf{9.68 \times 10^{-2}}$	1.17×10^{-1}	1.40×10^{-1}
IGD	Best	2.02×10^{-5}	1.96×10^{-5}	2.24×10^{-5}	$\mathbf{1.81 \times 10^{-5}}$
	Median	2.58×10^{-5}	$\mathbf{2.44 \times 10^{-5}}$	3.69×10^{-5}	2.61×10^{-5}
	Std.	2.80×10^{-5}	$\mathbf{1.62 \times 10^{-5}}$	3.99×10^{-5}	2.57×10^{-5}
HV	Best	1.24×10^{-3}	2.05×10^{-3}	$\mathbf{3.85 \times 10^{-3}}$	2.96×10^{-3}
	Median	$\mathbf{1.00 \times 10^{-3}}$	$\mathbf{1.00 \times 10^{-3}}$	$\mathbf{1.00 \times 10^{-3}}$	$\mathbf{1.00 \times 10^{-3}}$
	Std.	3.29×10^{-5}	1.56×10^{-4}	$\mathbf{3.98 \times 10^{-4}}$	3.52×10^{-4}

Appendix B.3. Tables in Experiment 4

Table A8. Experimental results for the Nikkei dataset by algebraic variation.

Metric		APG-SMOEA-4	APG-SMOEA-8	APG-SMOEA-9	APG-SMOEA-10	APG-SMOEA-11
GD	Best	$\mathbf{8.79} \times \mathbf{10^{-6}}$	1.30×10^{-5}	1.67×10^{-5}	1.07×10^{-5}	1.01×10^{-5}
	Median	2.92×10^{-5}	3.97×10^{-5}	3.96×10^{-5}	$\mathbf{2.36} \times \mathbf{10^{-5}}$	2.86×10^{-5}
	Std.	1.02×10^{-3}	1.98×10^{-3}	1.12×10^{-3}	1.28×10^{-3}	$\mathbf{2.55} \times \mathbf{10^{-5}}$
Spacing	Best	1.49×10^{-5}	1.99×10^{-5}	1.82×10^{-5}	1.25×10^{-5}	$\mathbf{7.15} \times \mathbf{10^{-6}}$
	Median	2.33×10^{-5}	2.28×10^{-5}	2.30×10^{-5}	2.31×10^{-5}	$\mathbf{2.27} \times \mathbf{10^{-5}}$
	Std.	5.28×10^{-5}	1.41×10^{-4}	9.78×10^{-5}	3.50×10^{-5}	$\mathbf{9.03} \times \mathbf{10^{-6}}$
MS	Best	1.88×10^{-2}	$\mathbf{4.71} \times \mathbf{10^{-2}}$	3.01×10^{-2}	2.30×10^{-2}	4.93×10^{-3}
	Median	4.26×10^{-3}	4.23×10^{-3}	4.24×10^{-3}	$\mathbf{4.27} \times \mathbf{10^{-3}}$	$\mathbf{4.27} \times \mathbf{10^{-3}}$
	Std.	$\mathbf{2.58} \times \mathbf{10^{-3}}$	6.23×10^{-3}	3.80×10^{-3}	3.06×10^{-3}	2.99×10^{-4}
Delta	Best	4.13×10^{-1}	4.19×10^{-1}	4.11×10^{-1}	4.24×10^{-1}	$\mathbf{3.93} \times \mathbf{10^{-1}}$
	Median	5.26×10^{-1}	5.16×10^{-1}	5.35×10^{-1}	5.20×10^{-1}	$\mathbf{5.04} \times \mathbf{10^{-1}}$
	Std.	1.50×10^{-1}	1.31×10^{-1}	1.21×10^{-1}	$\mathbf{9.23} \times \mathbf{10^{-2}}$	1.09×10^{-1}
IGD	Best	2.11×10^{-5}	$\mathbf{1.90} \times \mathbf{10^{-5}}$	2.21×10^{-5}	2.16×10^{-5}	$\mathbf{1.90} \times \mathbf{10^{-5}}$
	Median	3.00×10^{-5}	3.89×10^{-5}	3.79×10^{-5}	2.72×10^{-5}	$\mathbf{2.62} \times \mathbf{10^{-5}}$
	Std.	2.90×10^{-5}	3.74×10^{-5}	6.33×10^{-5}	$\mathbf{2.67} \times \mathbf{10^{-5}}$	5.84×10^{-5}
HV	Best	5.82×10^{-4}	7.28×10^{-4}	6.30×10^{-4}	$\mathbf{7.48} \times \mathbf{10^{-4}}$	3.19×10^{-4}
	Median	$\mathbf{2.88} \times \mathbf{10^{-4}}$	$\mathbf{2.88} \times \mathbf{10^{-4}}$	$\mathbf{2.88} \times \mathbf{10^{-4}}$	$\mathbf{2.88} \times \mathbf{10^{-4}}$	$\mathbf{2.88} \times \mathbf{10^{-4}}$
	Std.	6.68×10^{-5}	7.25×10^{-5}	6.27×10^{-5}	7.46×10^{-5}	$\mathbf{6.00} \times \mathbf{10^{-6}}$

Table A9. Experimental results for the FTSE dataset by adaptive parameters.

Metric		**APG-SMOEA-5**	**APG-SMOEA-12**	**APG-SMOEA-13**	**APG-SMOEA-14**
GD	Best	5.17×10^{-2}	8.76×10^{-2}	2.95×10^{-1}	$\mathbf{1.13 \times 10^{-2}}$
	Median	$\mathbf{8.12 \times 10^{-1}}$	9.48×10^{-1}	9.55×10^{-1}	8.26×10^{-1}
	Std.	1.72×10^{-1}	$\mathbf{1.52 \times 10^{-1}}$	1.78×10^{-1}	2.43×10^{-1}
Spacing	Best	9.19×10^{-4}	8.27×10^{-3}	2.41×10^{-3}	$\mathbf{2.73 \times 10^{-4}}$
	Median	$\mathbf{1.06 \times 10^{-2}}$	1.22×10^{-2}	1.47×10^{-2}	1.14×10^{-2}
	Std.	8.64×10^{-3}	8.80×10^{-3}	$\mathbf{2.55 \times 10^{-2}}$	9.52×10^{-3}
MS	Best	2.28×10^{0}	$\mathbf{2.41 \times 10^{0}}$	$\mathbf{2.41 \times 10^{0}}$	2.40×10^{0}
	Median	2.02×10^{0}	2.39×10^{0}	$\mathbf{2.40 \times 10^{0}}$	2.07×10^{0}
	Std.	3.96×10^{-1}	3.48×10^{-1}	$\mathbf{2.97 \times 10^{-1}}$	5.94×10^{-1}
Delta	Best	6.67×10^{-1}	6.98×10^{-1}	7.03×10^{-1}	$\mathbf{6.63 \times 10^{-1}}$
	Median	$\mathbf{8.31 \times 10^{-1}}$	8.35×10^{-1}	9.13×10^{-1}	8.51×10^{-1}
	Std.	$\mathbf{1.00 \times 10^{-1}}$	1.45×10^{-1}	1.94×10^{-1}	1.29×10^{-1}
IGD	Best	3.11×10^{-4}	5.53×10^{-4}	7.23×10^{-4}	$\mathbf{1.23 \times 10^{-4}}$
	Median	1.15×10^{-3}	1.36×10^{-3}	1.30×10^{-3}	$\mathbf{1.09 \times 10^{-3}}$
	Std.	2.65×10^{-4}	$\mathbf{2.08 \times 10^{-4}}$	2.13×10^{-4}	3.07×10^{-4}
HV	Best	3.84×10^{-1}	3.95×10^{-1}	3.97×10^{-1}	$\mathbf{3.99 \times 10^{-1}}$
	Median	3.67×10^{-1}	$\mathbf{3.74 \times 10^{-1}}$	3.67×10^{-1}	3.69×10^{-1}
	Std.	3.94×10^{-2}	$\mathbf{2.61 \times 10^{-2}}$	3.30×10^{-2}	6.40×10^{-2}

Table A10. Experimental results for the DAX dataset by adaptive parameters.

Metric		**APG-SMOEA-5**	**APG-SMOEA-12**	**APG-SMOEA-13**	**APG-SMOEA-14**
GD	Best	7.32×10^{-2}	$\mathbf{2.11 \times 10^{-4}}$	2.51×10^{-1}	8.33×10^{-3}
	Median	6.11×10^{-1}	6.83×10^{-1}	7.12×10^{-1}	$\mathbf{5.06 \times 10^{-1}}$
	Std.	1.63×10^{-1}	$\mathbf{1.43 \times 10^{-1}}$	2.16×10^{-1}	2.25×10^{-1}
Spacing	Best	1.64×10^{-3}	$\mathbf{1.15 \times 10^{-4}}$	4.84×10^{-4}	2.81×10^{-4}
	Median	1.03×10^{-2}	1.11×10^{-2}	1.01×10^{-2}	$\mathbf{9.66 \times 10^{-3}}$
	Std.	$\mathbf{6.55 \times 10^{-3}}$	9.92×10^{-3}	8.99×10^{-3}	1.26×10^{-2}
MS	Best	$\mathbf{1.92 \times 10^{0}}$	$\mathbf{1.92 \times 10^{0}}$	$\mathbf{1.92 \times 10^{0}}$	1.91×10^{0}
	Median	1.61×10^{0}	$\mathbf{1.91 \times 10^{0}}$	1.90×10^{0}	1.50×10^{0}
	Std.	3.71×10^{-1}	3.05×10^{-1}	$\mathbf{2.71 \times 10^{-1}}$	5.57×10^{-1}
Delta	Best	8.16×10^{-1}	7.76×10^{-1}	$\mathbf{7.43 \times 10^{-1}}$	8.18×10^{-1}
	Median	9.61×10^{-1}	$\mathbf{9.25 \times 10^{-1}}$	9.78×10^{-1}	9.56×10^{-1}
	Std.	$\mathbf{9.06 \times 10^{-2}}$	1.60×10^{-1}	2.01×10^{-1}	1.18×10^{-1}
IGD	Best	2.71×10^{-4}	$\mathbf{5.05 \times 10^{-5}}$	4.80×10^{-4}	2.24×10^{-4}
	Median	$\mathbf{7.08 \times 10^{-4}}$	7.18×10^{-4}	7.30×10^{-4}	7.14×10^{-4}
	Std.	1.09×10^{-4}	1.02×10^{-4}	$\mathbf{6.81 \times 10^{-5}}$	1.55×10^{-4}
HV	Best	1.81×10^{-1}	$\mathbf{2.03 \times 10^{-1}}$	1.86×10^{-1}	1.97×10^{-1}
	Median	1.59×10^{-1}	$\mathbf{1.70 \times 10^{-1}}$	1.66×10^{-1}	1.67×10^{-1}
	Std.	$\mathbf{1.72 \times 10^{-2}}$	2.64×10^{-2}	3.04×10^{-2}	3.62×10^{-2}

References

1. Liu, C.; Wu, S.; Li, R.; Jiang, D.; Wong, H.S. Self-supervised graph completion for incomplete multi-view clustering. *IEEE Trans. Knowl. Data Eng.* **2023**, *35*, 9394–9406. [CrossRef]
2. Ivanytska, A.; Zubyk, L.; Ivanov, D.; Domracheva, K. Study of Methods of Complex Data Analysis that Based on Machine Learning Technologies. In Proceedings of the 2019 IEEE International Conference on Advanced Trends in Information Theory (ATIT), Kyiv, Ukraine, 18–20 December 2019; pp. 332–335.
3. Li, S. A supply chain finance game model with order-to-factoring under blockchain. *Syst. Eng. Theory Pract.* **2023**, *43*, 3570–3586.
4. Lakhina, U.; Badruddin, N.; Elamvazuthi, I.; Jangra, A.; Huy, T.H.B.; Guerrero, J.M. An Enhanced Multi-Objective Optimizer for Stochastic Generation Optimization in Islanded Renewable Energy Microgrids. *Mathematics* **2023**, *11*, 2079. [CrossRef]

5. Guerrero, M.; Gil, C.; Montoya, F.G.; Alcayde, A.; Banos, R. Multi-objective evolutionary algorithms to find community structures in large networks. *Mathematics* **2020**, *8*, 2048. [CrossRef]
6. Deb, K.; Pratap, A.; Agarwal, S.; Meyarivan, T. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Trans. Evol. Comput.* **2002**, *6*, 182–197. [CrossRef]
7. Deb, K.; Jain, H. An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part I: Solving problems with box constraints. *IEEE Trans. Evol. Comput.* **2013**, *18*, 577–601. [CrossRef]
8. Beume, N.; Naujoks, B.; Emmerich, M. SMS-EMOA: Multiobjective selection based on dominated hypervolume. *Eur. J. Oper. Res.* **2007**, *181*, 1653–1669. [CrossRef]
9. Qi, Y.; Ma, X.; Liu, F.; Jiao, L.; Sun, J.; Wu, J. MOEA/D with adaptive weight adjustment. *Evol. Comput.* **2014**, *22*, 231–264. [CrossRef]
10. Xu, H.; Zeng, W.; Zhang, D.; Zeng, X. MOEA/HD: A multiobjective evolutionary algorithm based on hierarchical decomposition. *IEEE Trans. Cybern.* **2017**, *49*, 517–526. [CrossRef]
11. Xu, Y.; Ding, O.; Qu, R.; Li, K. Hybrid multi-objective evolutionary algorithms based on decomposition for wireless sensor network coverage optimization. *Appl. Soft Comput.* **2018**, *68*, 268–282. [CrossRef]
12. He, C.; Huang, S.; Cheng, R.; Tan, K.C.; Jin, Y. Evolutionary multiobjective optimization driven by generative adversarial networks (GANs). *IEEE Trans. Cybern.* **2020**, *51*, 3129–3142. [CrossRef] [PubMed]
13. Qian, W.; Liu, J.; Lin, Y.; Yang, L.; Zhang, J.; Xu, H.; Liao, M.; Chen, Y.; Chen, Y.; Liu, B. An improved MOEA/D algorithm for complex data analysis. *Wirel. Commun. Mob. Comput.* **2021**, *2021*, 6393638. [CrossRef]
14. Xu, M.; Zhang, M.; Cai, X.; Zhang, G. Adaptive neighbourhood size adjustment in MOEA/D-DRA. *Int. J. Bio-Inspired Comput.* **2021**, *17*, 14–23. [CrossRef]
15. Xu, H.; Xue, B.; Zhang, M. A duplication analysis-based evolutionary algorithm for biobjective feature selection. *IEEE Trans. Evol. Comput.* **2020**, *25*, 205–218. [CrossRef]
16. Xu, H.; Zeng, W.; Zeng, X.; Yen, G.G. An evolutionary algorithm based on Minkowski distance for many-objective optimization. *IEEE Trans. Cybern.* **2018**, *49*, 3968–3979. [CrossRef]
17. Xu, H.; Xue, B.; Zhang, M. Segmented initialization and offspring modification in evolutionary algorithms for bi-objective feature selection. In Proceedings of the 2020 Genetic and Evolutionary Computation Conference, Cancún, Mexico, 8–12 July 2020; pp. 444–452.
18. Xue, Y.; Cai, X.; Neri, F. A multi-objective evolutionary algorithm with interval based initialization and self-adaptive crossover operator for large-scale feature selection in classification. *Appl. Soft Comput.* **2022**, *127*, 109420. [CrossRef]
19. He, Y.; Aranha, C. Solving portfolio optimization problems using MOEA/D and levy flight. *Adv. Data Sci. Adapt. Anal.* **2020**, *12*, 2050005. [CrossRef]
20. Zhang, C.; Peng, Y. Stacking VAE and GAN for context-aware text-to-image generation. In Proceedings of the 2018 IEEE Fourth International Conference on Multimedia Big Data (BigMM), Xi'an, China, 13–16 September 2018; pp. 1–5.
21. Shiotani, M.; Iguchi, S.; Yamaguchi, K. Research on data augmentation for vital data using conditional GAN. In Proceedings of the 2022 IEEE 11th Global Conference on Consumer Electronics (GCCE), Osaka, Japan, 18–21 October 2022; pp. 344–345.
22. Yang, Y.; Wang, C.; Lin, L. Regional Style Transfer Based on Partial Convolution Generative Adversarial Network. In Proceedings of the 2020 Chinese Automation Congress (CAC), Shanghai, China, 6–8 November 2020; pp. 5234–5239.
23. Leung, M.F.; Wang, J. A collaborative neurodynamic approach to multiobjective optimization. *IEEE Trans. Neural Netw. Learn. Syst.* **2018**, *29*, 5738–5748. [CrossRef]
24. Wang, Z.; Yao, S.; Li, G.; Zhang, Q. Multiobjective Combinatorial Optimization Using a Single Deep Reinforcement Learning Model. *IEEE Trans. Cybern.* **2023**, *in press*.
25. Huang, W.; Zhang, Y.; Li, L. Survey on multi-objective evolutionary algorithms. *J. Phys. Conf. Ser.* **2019**, *1288*, 012057. [CrossRef]
26. Farina, M.; Amato, P. On the optimal solution definition for many-criteria optimization problems. In Proceedings of the 2002 Annual Meeting of the North American Fuzzy Information Processing Society Proceedings, NAFIPS-FLINT 2002 (Cat. No. 02TH8622), New Orleans, LA, USA, 27–29 June 2002; pp. 233–238.
27. El-Nemr, M.; Afifi, M.; Rezk, H.; Ibrahim, M. Finite element based overall optimization of switched reluctance motor using multi-objective genetic algorithm (NSGA-II). *Mathematics* **2021**, *9*, 576. [CrossRef]
28. Yu, G.; Chai, T.; Luo, X. Two-level production plan decomposition based on a hybrid MOEA for mineral processing. *IEEE Trans. Autom. Sci. Eng.* **2012**, *10*, 1050–1071. [CrossRef]
29. Tian, Y.; Cheng, R.; Zhang, X.; Cheng, F.; Jin, Y. An indicator-based multiobjective evolutionary algorithm with reference point adaptation for better versatility. *IEEE Trans. Evol. Comput.* **2017**, *22*, 609–622. [CrossRef]
30. Sun, Y.; Yen, G.G.; Yi, Z. IGD indicator-based evolutionary algorithm for many-objective optimization problems. *IEEE Trans. Evol. Comput.* **2018**, *23*, 173–187. [CrossRef]
31. Xu, H.; Zeng, W.; Zeng, X.; Yen, G.G. A polar-metric-based evolutionary algorithm. *IEEE Trans. Cybern.* **2020**, *51*, 3429–3440. [CrossRef] [PubMed]
32. Ravber, M.; Mernik, M.; Črepinšek, M. The impact of quality indicators on the rating of multi-objective evolutionary algorithms. *Appl. Soft Comput.* **2017**, *55*, 265–275. [CrossRef]
33. Zhang, Q.; Li, H. MOEA/D: A multiobjective evolutionary algorithm based on decomposition. *IEEE Trans. Evol. Comput.* **2007**, *11*, 712–731. [CrossRef]

34. Wang, Z.; Zhang, Q.; Zhou, A.; Gong, M.; Jiao, L. Adaptive replacement strategies for MOEA/D. *IEEE Trans. Cybern.* **2015**, *46*, 474–486. [CrossRef]
35. Wang, W.X.; Li, K.S.; Tao, X.Z.; Gu, F.H. An improved MOEA/D algorithm with an adaptive evolutionary strategy. *Inf. Sci.* **2020**, *539*, 1–15. [CrossRef]
36. Falcón-Cardona, J.G.; Coello, C.A.C. Indicator-based multi-objective evolutionary algorithms: A comprehensive survey. *ACM Comput. Surv. (CSUR)* **2020**, *53*, 1–35. [CrossRef]
37. Lotfi, S.; Karimi, F. A Hybrid MOEA/D-TS for solving multi-objective problems. *J. AI Data Min.* **2017**, *5*, 183–195.
38. Abdi, Y.; Feizi-Derakhshi, M.R. Hybrid multi-objective evolutionary algorithm based on search manager framework for big data optimization problems. *Appl. Soft Comput.* **2020**, *87*, 105991. [CrossRef]
39. Silva, Y.L.T.; Herthel, A.B.; Subramanian, A. A multi-objective evolutionary algorithm for a class of mean-variance portfolio selection problems. *Expert Syst. Appl.* **2019**, *133*, 225–241. [CrossRef]
40. Jabbar, A.; Li, X.; Omar, B. A survey on generative adversarial networks: Variants, applications, and training. *ACM Comput. Surv. (CSUR)* **2021**, *54*, 1–49. [CrossRef]
41. Goodfellow, I.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A.; Bengio, Y. Generative adversarial networks. *Commun. ACM* **2020**, *63*, 139–144. [CrossRef]
42. Gui, J.; Sun, Z.; Wen, Y.; Tao, D.; Ye, J. A review on generative adversarial networks: Algorithms, theory, and applications. *IEEE Trans. Knowl. Data Eng.* **2021**, *35*, 3313–3332. [CrossRef]
43. Goodfellow, I.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A.; Bengio, Y. Generative adversarial nets. *Adv. Neural Inf. Process. Syst.* **2014**, *27*.
44. Goodfellow, I.J. On distinguishability criteria for estimating generative models. *arXiv* **2014**, arXiv:1412.6515.
45. Brophy, E.; Wang, Z.; She, Q.; Ward, T. Generative adversarial networks in time series: A systematic literature review. *ACM Comput. Surv.* **2023**, *55*, 1–31. [CrossRef]
46. Aggarwal, A.; Mittal, M.; Battineni, G. Generative adversarial network: An overview of theory and applications. *Int. J. Inf. Manag. Data Insights* **2021**, *1*, 100004. [CrossRef]
47. Dong, H.; Yu, S.; Wu, C.; Guo, Y. Semantic image synthesis via adversarial learning. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 5706–5714.
48. Ma, L.; Jia, X.; Sun, Q.; Schiele, B.; Tuytelaars, T.; Van Gool, L. Pose guided person image generation. In Proceedings of the 31st Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA, 4–9 December 2017; Volume 30.
49. Fahim Sikder, M. Bangla handwritten digit recognition and generation. In Proceedings of the International Joint Conference on Computational Intelligence: IJCCI 2018, Birulia, Bangladesh, 14–15 December 2018; Springer: Berlin/Heidelberg, Germany, 2020; pp. 547–556.
50. Kelkar, V.A.; Gotsis, D.S.; Brooks, F.J.; Prabhat, K.; Myers, K.J.; Zeng, R.; Anastasio, M.A. Assessing the ability of generative adversarial networks to learn canonical medical image statistics. *IEEE Trans. Med Imaging* **2023**, *42*, 1799–1808. [CrossRef]
51. Yin, X.; Yu, X.; Sohn, K.; Liu, X.; Chandraker, M. Towards large-pose face frontalization in the wild. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 Octobr 2017; pp. 3990–3999.
52. Odena, A.; Olah, C.; Shlens, J. Conditional image synthesis with auxiliary classifier gans. In Proceedings of the International Conference on Machine Learning, PMLR, Sydney, Australia, 6–11 August 2017; pp. 2642–2651.
53. Hamada, K.; Tachibana, K.; Li, T.; Honda, H.; Uchida, Y. Full-body high-resolution anime generation with progressive structure-conditional generative adversarial networks. In Proceedings of the European Conference on Computer Vision (ECCV) Workshops, Munich, Germany, 8–14 September 2018.
54. Wang, Z.; She, Q.; Ward, T.E. Generative adversarial networks in computer vision: A survey and taxonomy. *ACM Comput. Surv. (CSUR)* **2021**, *54*, 1–38. [CrossRef]
55. He, Z.; Zuo, W.; Kan, M.; Shan, S.; Chen, X. Attgan: Facial attribute editing by only changing what you want. *IEEE Trans. Image Process.* **2019**, *28*, 5464–5478. [CrossRef] [PubMed]
56. Ehsani, K.; Mottaghi, R.; Farhadi, A. Segan: Segmenting and generating the invisible. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 6144–6153.
57. Denton, E.L.; Birodkar, V. Unsupervised learning of disentangled representations from video. In Proceedings of the 31st Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA, 4–9 December 2017; Volume 30.
58. Li, C.; Wand, M. Precomputed real-time texture synthesis with markovian generative adversarial networks. In Proceedings of the Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, 11–14 October 2016; Proceedings, Part III 14; Springer: Berlin/Heidelberg, Germany, 2016; pp. 702–716.
59. Gao, N.; Xue, H.; Shao, W.; Zhao, S.; Qin, K.K.; Prabowo, A.; Rahaman, M.S.; Salim, F.D. Generative adversarial networks for spatio-temporal data: A survey. *ACM Trans. Intell. Syst. Technol. (TIST)* **2022**, *13*, 1–25. [CrossRef]
60. Yu, L.; Zhang, W.; Wang, J.; Yu, Y. Seqgan: Sequence generative adversarial nets with policy gradient. In Proceedings of the AAAI Conference on Artificial Intelligence, San Francisco, CA, USA, 4–9 February 2017; Volume 31.
61. Liu, S.; Jiang, H.; Wu, Z.; Li, X. Data synthesis using deep feature enhanced generative adversarial networks for rolling bearing imbalanced fault diagnosis. *Mech. Syst. Signal Process.* **2022**, *163*, 108139. [CrossRef]

62. Lu, S.; Dou, Z.; Jun, X.; Nie, J.Y.; Wen, J.R. Psgan: A minimax game for personalized search with limited and noisy click data. In Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, Paris, France, 21–25 July 2019; pp. 555–564.
63. Siddique, N.; Adeli, H. *Computational Intelligence: Synergies of Fuzzy Logic, Neural Networks and Evolutionary Computing*; John Wiley & Sons: Hoboken, NJ, USA, 2013.
64. Zhang, J.; Liang, C.; Lu, Q. A novel small-population genetic algorithm based on adaptive mutation and population entropy sampling. In Proceedings of the 2008 7th World Congress on Intelligent Control and Automation, Changsha, China, 4–8 July 2008; pp. 8738–8742.
65. Li, H.; Zhang, Q. Multiobjective optimization problems with complicated Pareto sets, MOEA/D and NSGA-II. *IEEE Trans. Evol. Comput.* **2008**, *13*, 284–302. [CrossRef]
66. Zhang, Q.; Liu, W.; Tsang, E.; Virginas, B. Expensive multiobjective optimization by MOEA/D with Gaussian process model. *IEEE Trans. Evol. Comput.* **2009**, *14*, 456–474. [CrossRef]
67. Yang, H.; Li, Y.; Yang, L.; Wu, Q. An improved particle swarm optimization algorithm based on entropy and fitness of particles. In Proceedings of the 2020 12th International Conference on Measuring Technology and Mechatronics Automation (ICMTMA), Phuket, Thailand, 28–29 February 2020; pp. 492–496.
68. Lin, C.; Xu, C.; Luo, D.; Wang, Y.; Tai, Y.; Wang, C.; Li, J.; Huang, F.; Fu, Y. Learning salient boundary feature for anchor-free temporal action localization. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 20–25 June 2021; pp. 3320–3329.
69. Ladosz, P.; Weng, L.; Kim, M.; Oh, H. Exploration in deep reinforcement learning: A survey. *Inf. Fusion* **2022**, *85*, 1–22. [CrossRef]