

# Multi-population GAN Training: Analyzing Co-Evolutionary Algorithms

Walter P. Casas

ITIS UMA, University of Malaga  
Spain  
walterpcasas@gmail.com

Jamal Toutouh

ITIS UMA, University of Malaga  
Spain  
jamal@uma.es

## Abstract

Generative adversarial networks (GANs) are powerful generative models but remain challenging to train due to pathologies such as mode collapse and instability. Recent research has explored co-evolutionary approaches, in which populations of generators and discriminators are evolved, as a promising solution. This paper presents an empirical analysis of different coevolutionary GAN training strategies, focusing on the impact of selection and replacement mechanisms. We compare  $(\mu, \lambda)$ ,  $(\mu+\lambda)$  with elitism, and  $(\mu+\lambda)$  with tournament selection coevolutionary schemes, along with a non-evolutionary population based multi-generator multi-discriminator GAN baseline, across both synthetic low-dimensional datasets (blob and gaussian mixtures) and an image-based benchmark (MNIST). Results show that full generational replacement, i.e.,  $(\mu, \lambda)$ , consistently outperforms in terms of both sample quality and diversity, particularly when combined with larger offspring sizes. In contrast, elitist approaches tend to converge prematurely and suffer from reduced diversity. These findings highlight the importance of balancing exploration and exploitation dynamics in coevolutionary GAN training and provide guidance for designing more effective population-based generative models.

## CCS Concepts

- Computing methodologies → Unsupervised learning; Bio-inspired approaches; Neural networks.

## Keywords

Generative Adversarial Networks, Coevolutionary Algorithms, Diversity

## ACM Reference Format:

Walter P. Casas and Jamal Toutouh. 2025. Multi-population GAN Training: Analyzing Co-Evolutionary Algorithms. In . ACM, New York, NY, USA, 9 pages. <https://doi.org/10.1145/nnnnnnnn.nnnnnnn>

## 1 Introduction

Generative Adversarial Networks (GANs) have emerged as powerful generative models, demonstrating success in image synthesis, data augmentation, and unsupervised representation learning [28].

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from [permissions@acm.org](mailto:permissions@acm.org).

*Conference'17, Washington, DC, USA*

© 2025 Copyright held by the owner/author(s). Publication rights licensed to ACM.  
ACM ISBN 978-x-xxxx-xxxx-x/YYYY/MM  
<https://doi.org/10.1145/nnnnnnnn.nnnnnnn>

A GAN consists of two artificial neural networks (ANN), a generator ( $g$ ) and a discriminator ( $d$ ), that are trained simultaneously in a competitive setting. The generator aims to produce samples that resemble real training data, while the discriminator attempts to distinguish between real and synthesized samples [21].

The training process is formulated as a min-max optimization problem [27]. The loss functions are designed so that the generator is rewarded for successfully fooling the discriminator, while the discriminator is rewarded for correctly distinguishing between real and fake data samples.

Under ideal conditions, GAN training converges to an equilibrium where the generator produces samples that are indistinguishable from real data. However, in practice, achieving stable convergence remains a significant challenge. Vanishing gradient [6], mode collapse [7] or discriminator collapse [29] are some of the GAN training pathologies that can arise during the training.

Several approaches have been proposed to overcome GAN pathologies. Among others, researchers have proposed multi-generator and multi-discriminator architectures, evolutionary-based GAN training, and co-evolutionary approaches. Multi-generator GANs introduce multiple competing generators to enhance sample diversity and mitigate mode collapse, while multi-discriminator frameworks leverage multiple critics to provide more refined and distributed adversarial feedback. Evolutionary algorithms (EAs) have been applied to GAN training to optimize hyperparameters, loss functions, and network architectures through mechanisms such as selection, mutation, and recombination. Recently, co-evolutionary algorithms (Co-EAs) have emerged as a promising paradigm, applying population-based co-evolution to GAN training, where both populations of generators and of discriminators co-adapt over successive iterations. These methods have demonstrated improvements in stability, diversity, and training efficiency.

This work aims to analyze how different selection, replacement, and population dynamics influence Co-EA GAN training stability, diversity, and convergence. Thus, the main contribution of this work is a systematic evaluation of different coevolutionary strategies for GAN training, focusing on how population dynamics, selection mechanisms, and replacement strategies impact both quality and diversity. To guide this analysis, we consider the following research questions: **RQ1:** How do different coevolutionary strategies—particularly selection and replacement mechanisms—impact the quality and diversity of GAN-generated samples? **RQ2:** To what extent does the balance between exploration and exploitation, as controlled by population and offspring sizes, affect the performance and stability of GAN training? **RQ3:** Can a simple population

based multi-generator multi-discriminator (MG-MD) GAN without selection or replacement be competitive with coevolutionary approaches?

The remainder of the paper is organized as follows. Section 2 introduces the background and related work. The evaluated Co-EA GAN training approaches are described in Section 3. Section 4 describes the experimental methodology and Section 5 presents the experimental results and discussions. Section 6 discusses the main conclusions and future work.

## 2 Related Works

Generative Adversarial Networks (GANs) have been used across a wide variety of applications, from image and video synthesis to data enhancement [1]. Despite their remarkable success, training GANs remains complex due to the intricate interaction between the generator and the discriminator, which can lead to undesirable behaviors. This complex interaction frequently results in training pathologies like mode collapse and instability, making these models particularly difficult to train.

Several methods integrate Evolutionary Computation (EC) for GANs to mitigate their training pathologies. One representative example is Evolutionary GAN (EGAN) [47] which employs multiple loss functions, each assigned to a different generator, and selects the best performing generator, introducing diversity.

Another notable approach is Multi-objective Evolutionary GAN (MO-EGAN) [10], which proposes a framework that simultaneously optimizes conflicting criteria such as fidelity and diversity. In MO-EGAN, a multi-objective evolutionary algorithm is used to evolve the generator, resulting in a more balanced strategy that addresses inherent challenges and enhances both the robustness and quality of the generated outputs.

An additional contribution is Differential-evolution-based GAN (DE-GAN) [52]. This approach applies differential evolution to explore the parameter space in GAN training, specifically targeting edge detection. DE-GAN addresses common training pathologies optimizing the GAN's parameters, demonstrating the potential of evolutionary strategies to enhance performance in specialized tasks.

Other approaches have shown promise in addressing inherent adversarial challenges. Adversarial Evolutionary Learning (AEL), implemented as competitive CoEA [51], has proven effective for search, optimization, design and modeling [3, 32, 33, 37]. Its applications include board games [31], video game playing [26, 38], social science games [8], software engineering tasks as coevolving programs and unit tests [4, 5, 49] or differentiating correct from incorrect behavior [11], and various cybersecurity problems [25, 34, 36].

As GANs represent a type of adversarial learning, some authors have successfully applied Competitive Co-EA to train GANs. CO-EGAN [13, 14] takes to a coevolutionary approach to jointly evolve the architectures of generator and discriminator, and t-SNE visualization analysis suggest convergence to high quality model [15]. Numerous extensions of this idea include multi objective variants, based on NSGA-II [10], quality diversity methods. Differential evolution approaches and cooperative CoEA that evolve both generators and discriminators for multi-objective optimization. Other works replace the generator with an expression-based population. Few works explore spatial distributed CoEA for GAN training, where

adversaries are arranged within a spatial neighborhood or grid. Spatial organization can provide advantages in diversity maintenance, hyperparameter exploration and fine-grained interactions during training [23, 40, 41, 43–45].

Most evolutionary GANs applications focus on computer vision tasks such as hyperspectral image classification [9], abnormal electrocardiogram classification [48], data augmentation for cardiac magnetic resonance image [19], or COVID-19 infection segmentation [18, 22, 40]. Beyond vision, evolutionary GAN have been employed for natural language generation [39], sequential data imputation [12], and evolving the GAN architectures of generators and discriminators [16, 20, 50].

## 3 GAN Training Methods Evaluated

This section introduces the standard GAN and the multi-population GAN training approaches evaluated in this paper: multi-generator multi-discriminator GAN (MG-MD GAN),  $(\mu, \lambda)$  Co-EA GAN, and  $(\mu + \lambda)$  Co-EA GAN.

Standard GAN trains a single pair of artificial neural networks (ANNs) named generator  $G$  and discriminator  $D$  using adversarial learning. In this framework, during the iterative training process, the generator  $G$  attempts to produce realistic samples to deceive the discriminator, while  $D$  aims to distinguish between real samples from the training data set and produced samples created by  $G$ . Equation 1 shows the mathematical formulation proposed by Goodfellow et al. [21]. Here,  $p_{\text{real}}$  denotes the real data distribution and  $p_z$  is a prior distribution over the latent space.

$$\min_G \max_D V(D, G) = \mathbb{E}_{x \sim p_{\text{real}}} [\log D(x)] + \mathbb{E}_{z \sim p_z} [\log(1 - D(G(z)))] \quad (1)$$

MG-MD GAN incorporates multiple generators and discriminators organized as fixed populations. However, unlike evolutionary methods, there is no selection or replacement during training. During the training process, for each iteration, the generators and discriminators are randomly coupled. Then, each generator-discriminator pair is trained independently using gradient descent according to Equation 1 for a fixed number  $n_t$  of epochs. After a given stop criterion, at the end of the training, the best generator and discriminator are selected based on a predefined quality metric. This setup allows isolating the impact of population diversity from evolutionary dynamics [42].

In  $(\mu, \lambda)$  Co-EA GAN, evolutionary dynamics are introduced through a  $(\mu, \lambda)$  Co-EA. Two evolving populations are maintained: one for generators and one for discriminators. At each generation,  $\lambda$  individuals are selected from each population using a fitness-based selection process, e.g., tournament selection, (Algorithm 1, lines 6–7). These selected individuals form the offspring populations  $g^*$  and  $d^*$ .

Each offspring generator is randomly paired with a discriminator from the offspring discriminator pool (lines 6–12) and trained for  $n_t$  epochs using gradient descent. After training, the offspring are added to the current populations, which in the case of  $(\mu, \lambda)$  is empty, (lines 14–15), and fitness is re-evaluated (line 16).

**Algorithm 1** Co-EA training

**Input:**  $T_B$ : Computational budget (training epochs),  $p_{real}$ : Training dataset,  $B_s$ : Batch size,  $\theta_g, \theta_d$ : Initial generator and discriminator parameters,  $\mu$ : Population size,  $\lambda$ : Offspring size,  $\tau$ : Selection parameters,  $n_e$ : Number of evaluation batches,  $n_t$ : Number of training epochs per generator-discriminator couple

**Return:**  $g, d$ : Trained generator and discriminator

---

```

1:  $g \leftarrow \text{initializePopulation}(\theta_g)$                                 ▷ Initialize population g
2:  $d \leftarrow \text{initializePopulation}(\theta_d)$                             ▷ Initialize population d
3:  $t \leftarrow \left\lfloor \frac{T_B}{n_t \lambda} \right\rfloor$                       ▷ Compute the number of generations to perform
4:  $\mathcal{L}_{g,d} \leftarrow \text{evaluate}(g, d, n_e)$                            ▷ Evaluate populations
5: for  $i = 1$  to  $t$  do                                                 ▷ Loop over generations
6:    $g^* \leftarrow \text{select}(\lambda, \tau, \mathcal{L}_{g,d})$                          ▷ Selection to get offspring population  $g^*$ 
7:    $d^* \leftarrow \text{select}(\lambda, \tau, \mathcal{L}_{g,d})$                          ▷ Selection to get offspring population  $d^*$ 
8:    $d' \leftarrow d^*$                                                        ▷ Initialize the set of disc. to couple with gen.
9:   for  $g \in g^*$  do                                                 ▷ Loop over  $g$  in the offspring  $g^*$ 
10:     $d \leftarrow \text{pickOneRandomly}(d')$                                      ▷ Select a discriminator for  $g$ 
11:     $d' \leftarrow d' - \{d\}$                                               ▷ Remove the selected  $d$  from the set
12:     $g, d \leftarrow \text{train}(g, d, n_t, p_{real}, B_s, \theta_g, \theta_d)$           ▷ Train
13:  end for
14:   $g \leftarrow g \cup g^*$                                               ▷ Add offspring to population
15:   $d \leftarrow d \cup d^*$                                               ▷ Add offspring to population
16:   $\mathcal{L}_{g,d} \leftarrow \text{evaluate}(g, d, n_e)$                            ▷ Evaluate populations
17:   $g \leftarrow \text{updatePopulation}(g)$                                      ▷ Apply replacement
18:   $d \leftarrow \text{updatePopulation}(d)$                                      ▷ Apply replacement
19: end for
20:  $g \leftarrow \text{selectBestGenerator}(g)$ 
21:  $d \leftarrow \text{selectBestDiscriminator}(d)$ 
22: return  $g, d$                                                        ▷ Return  $g, d$  trained SSL-GAN

```

---

In the  $(\mu, \lambda)$  strategy, the final population update step (lines 17–18) discards the parent population and retains only the  $\mu$  best individuals from the  $\lambda$  offspring, enforcing full generational replacement and promoting exploration.

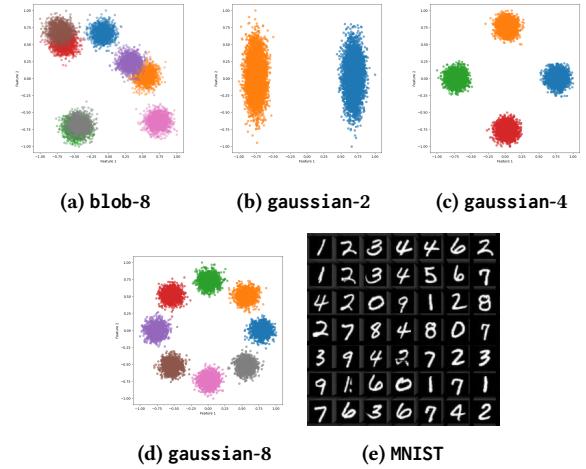
The  $(\mu+\lambda)$  Co-EA GAN introduces elitism by modifying the replacement strategy. Here, the population update selects the top  $\mu$  individuals from the union of parents and offspring. This approach preserves high-performing individuals from previous generations, which encourages exploitation of learned solutions while maintaining evolutionary innovation.

This variant shares the same initialization, selection, training, and evaluation phases as the  $(\mu, \lambda)$  Co-EA GAN (Algorithm 1), but differs in how the replacement is performed in lines 17–18. The function `updatePopulation()` is adapted to select the  $\mu$  best individuals from  $g \cup g^*$  and  $d \cup d^*$  rather than just the offspring.

## 4 Experimental Setup

The experimental evaluation is performed on three datasets: a mixture of eight 2D Gaussian distributions randomly arranged, referred to as blob; synthetic datasets comprising two, four, and ten 2D Gaussian modes arranged uniformly in a circle, referred to as gaussian-2, gaussian-4, and gaussian-8, respectively; and the standard Modified National Institute of Standards and Technology database, MNIST [17]. Figure 1 provides visual examples of the used datasets.

Both blob and gaussian datasets consist of 2D vectors within the range  $[-1, 1]$  and are divided into ten clusters (i.e., Gaussian distributions), forming a total of 10,000 training samples and 1,000 test samples. Although simple in dimensionality, these datasets are widely adopted in generative modeling research due to the



**Figure 1: Representation of the datasets applied in our empirical analysis.**

challenges they pose—particularly in terms of capturing diversity without collapsing into a subset of the modes.

Although both the gaussian and blob datasets are composed of mixtures 2D Gaussians, they present different challenges from a generative modeling point of view. The gaussian dataset features a number of well-separated Gaussian modes arranged symmetrically in a circular pattern (see figures 1b, 1c, and 1d). While this structure makes the distribution visually interpretable, it also imposes a stricter requirement on the generator to produce samples across all distinct modes. This characteristic makes gaussian particularly susceptible to mode collapse, where the generator may focus on a subset of the modes and ignore the rest.

On the other hand, the blob dataset contains eight Gaussian modes positioned randomly in the feature space, often with some degree of overlap between them (see Figure 1a). This lack of clear separation reduces the penalty for missing certain regions of the data distribution, making the generation task comparatively easier. Since the modes are not uniformly distributed, even generators that fail to model all components accurately may still produce samples that appear plausible under the data distribution.

The MNIST dataset comprises grayscale images of handwritten digits from 0 to 9, each with a resolution of 28×28 pixels. The training partition includes 60,000 samples, while the test set contains 10,000 images. Due to its complexity and high-dimensional nature, MNIST serves as a standard benchmark for evaluating the generative quality and diversity of image-based models.

In our experiments, the performance of the evaluated methods is evaluated in terms of sample quality and diversity. For blob and gaussian datasets, quality is assessed using the Wasserstein Distance ( $W_D$ ) [46], which measures distributional discrepancies in a 2D space. Diversity is quantified through mode coverage [30] and cluster-based entropy [2], which respectively capture how many Gaussian modes are represented and how evenly samples are distributed among them, respectively.

In the MNIST experiments, sample quality is measured using the Fréchet Inception Distance (FID) [24], a standard metric for visual realism based on deep feature statistics. Diversity is evaluated using Total Variation Distance (TVD) [21] between label distributions of real and generated samples, along with mode coverage over the ten digit classes.

The experiments are conducted using two distinct architectures tailored to the datasets under study. A simple multilayer perceptron (MLP)-based architecture is used to address blob and gaussian data generation and a convolutional neural network (CNN) is used to produce MNIST.

In the case of blob and gaussian, the generator is a three-layer MLP that transforms a latent vector into a 2D point. It includes a hidden layer with 128 ReLU units and a final tanh activation to constrain outputs to  $[-1, 1]$ . The discriminator is a three-layer MLP with a hidden layer with 128 LeakyReLU units and a sigmoid unit outputs a binary classification (real/fake).

For MNIST, the generator begins with a fully connected layer mapping the latent vector into a  $7 \times 7$  feature map, which is then upsampled through two transposed convolutional layers with ReLU activations and batch normalization, producing a final  $28 \times 28$  grayscale image. The discriminator consists of four convolutional blocks that downsample the input, followed by a sigmoid-based binary classifier.

The experimental design focuses on assessing the impact of key factors in a co-evolutionary algorithm: population sizes, offspring, and selection/replacement strategy. For the  $(\mu+\lambda)$  variant, we applied elitism and tournament selection, testing population sizes  $\mu \in \{3, 5, 7\}$ , and offspring sizes  $\lambda \in \{1, \lceil \mu/2 \rceil, \mu\}$ . The  $(\mu+\lambda)$  variant that applies elitism is named  $(\mu+\lambda)_E$  and the variant that uses tournament selection is referred as  $(\mu+\lambda)_T$ . For the  $(\mu, \lambda)$  strategy, we also employed tournament selection, with  $\mu \in \{3, 5, 7\}$  and  $\lambda \in \{\mu, \lceil 1.5\mu \rceil, 2\mu\}$ . The choice of  $\lambda$  values in each case was made to ensure that the effective selective pressure when generating the next population remained comparable between both methods. The static coevolution baseline (without selection or replacement) was evaluated using fixed population sizes of 3, 5, and 7.

A set of preliminary experiments were performed to confirm the main hyper-parameters used to train the GANs proposed by Sedeño et al. [35]. Generators and discriminators apply the Adam optimizer with the same learning rate 0.0003. The batch size is 100 samples for blob and gaussian and 600 for MNIST. The total training epochs  $T_B$  is set to  $250\lambda$  and the number of training epochs  $n_t$  when coupling generator-discriminator is 5.

## 5 Experimental Results

In this section, we present the results of the experiments, split by type of datasets, and discusses the research questions. Thirty independent runs were performed in all the cases.

### 5.1 Results on blob and gaussian

Figure 2 presents the distribution of  $W_D$  scores across all evaluated methods considering the different  $\mu$  and  $\lambda$  on the gaussian and blob datasets.

In general, Figure 2 shows that among the evaluated datasets, gaussian-4 consistently exhibits the highest  $W_D$  values, which

indicates it is the most challenging setting for all methods. This can be attributed to the structured and symmetric arrangement of modes in the gaussian dataset, which imposes strong constraints on the generator to uniformly cover all regions. Any mode collapse or imbalance in sample distribution results in a significant penalty under  $W_D$ . In contrast, the blob dataset yields the lowest  $W_D$  scores across methods, reflecting its relative ease for generative modeling.

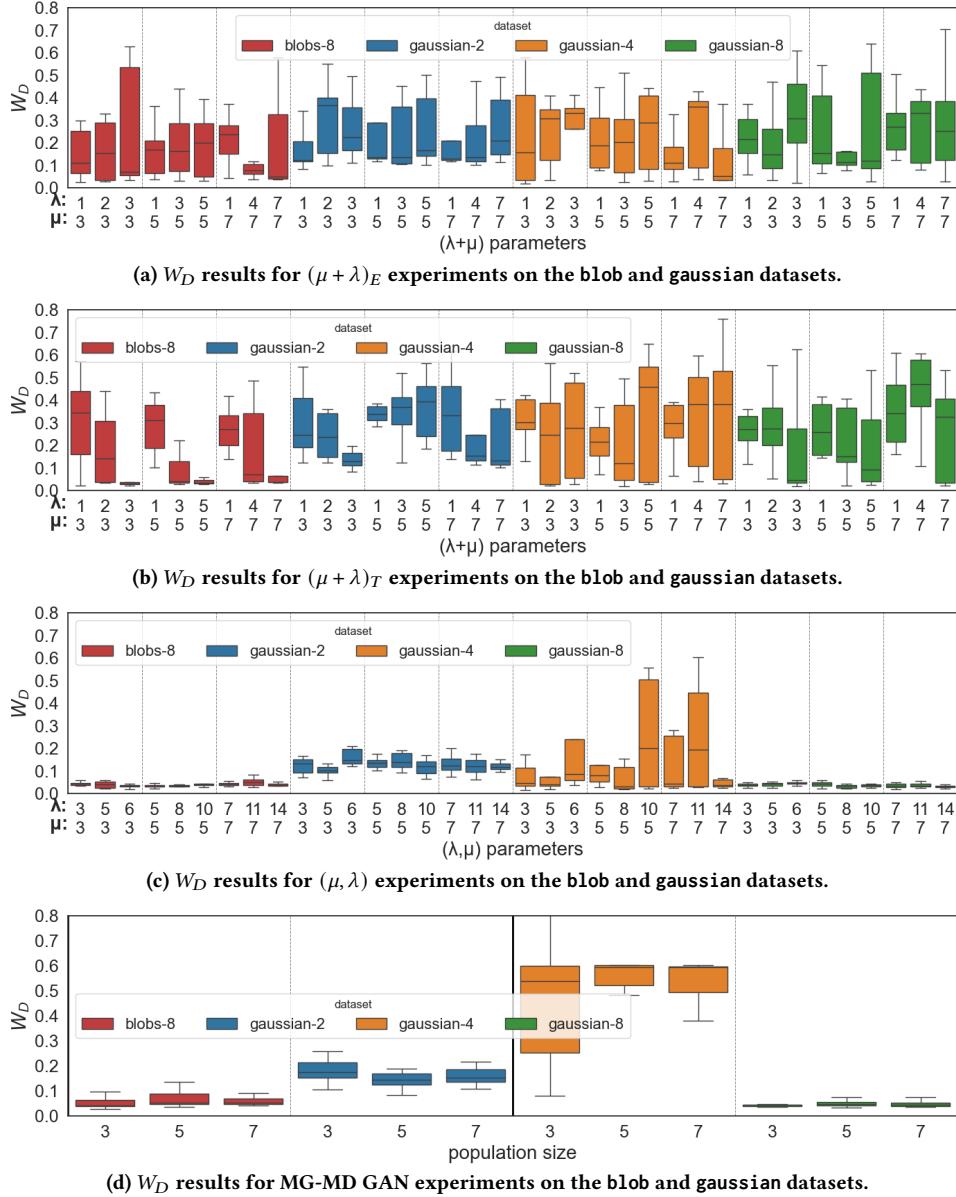
When comparing the  $(\mu+\lambda)$  variants, Figure 2 shows that  $(\mu+\lambda)_E$  underperforms  $(\mu+\lambda)_T$ . Elitism reduces exploratory pressure by preserving top individuals across generations, which can lead to early convergence and limited diversity in the population. This lack of diversity appears to hinder the generator's ability to capture all data modes, especially in more structured distributions like gaussian. Moreover, in elitist settings, increasing the population or offspring size does not lead to quality improvements, suggesting that the preserved elite dominates the evolutionary dynamics, reducing the effect of new individuals. In contrast, tournament-based selection shows a clear improvement in  $W_D$  scores as the offspring size increases. This can be attributed to the higher selective pressure and greater exploration of the search space enabled by larger offspring pools. While increasing the population size also has a positive effect, the results suggest that offspring size plays a more critical role, likely because it directly controls the volume of candidate solutions explored in each generation.

The  $(\mu, \lambda)$  variation provides the most competitive  $W_D$  results. It also exhibits greater robustness, as reflected by narrower interquartile ranges in Figure 2. While improvements are observed with increasing population and offspring sizes, these gains are generally limited and not always statistically significant according to the Wilcoxon rank-sum test. A key factor behind the effectiveness of  $(\mu, \lambda)$  is its stronger capacity to preserve diversity across generations, as offspring completely replace the parent population. This strategy encourages broader exploration of the search space and reduces the risk of premature convergence—a phenomenon previously discussed in the context of evolutionary optimization and coevolutionary GANs [23].

The baseline, MG-MD GAN, without selection or replacement also shows competitive results, outperforming  $(\mu+\lambda)_E$  and approaching the best  $(\mu+\lambda)_T$  results. These findings highlight the sensitivity of coevolutionary GAN training to the choice of evolutionary strategy and underscore the importance of balancing exploration and exploitation dynamics in population-based training.

To analyze the diversity of the generated samples, we report results using two complementary metrics: cluster-based entropy, which measures how evenly the generator distributes samples across clusters, and mode coverage, which counts how many distinct Gaussian modes are captured by the generated distribution. Tables 1 and 2 summarize the cluster-based entropy for  $(\mu+\lambda)$  and  $(\mu, \lambda)$  variants, respectively, and Tables 3 and 4 present the corresponding mode coverage results. MG-MD GAN baseline diversity metrics are also reported for comparison. These tables present median values and interquartile range (iqr) on 30 independent runs. Due to space limitations, we discuss the diversity for the datasets that showed the extreme cases in terms of  $W_D$ , i.e., blob-8 (best  $W_D$  results) and gaussian-4 (worst  $W_D$  results).

In terms of cluster-based entropy, the  $(\mu, \lambda)$  strategy exhibits the most stable and consistent behavior across both blob-8 and



**Figure 2: Influence of  $\mu$  and  $\lambda$  size in  $W_D$  on the blob and gaussian datasets.**

gaussian-4 datasets. As shown in Table 2, it achieves median entropy values close to 2.0 on blob-8 with very low interquartile ranges (e.g., 1.973 (0.049) for  $\mu = 5, \lambda = 10$ ), indicating a balanced spread of generated samples across all clusters. On the more challenging gaussian-4, entropy values are higher and more dispersed, but still consistently outperform the elitist  $(\mu+\lambda)_E$  approach and match or surpass tournament-based variants.

The  $(\mu+\lambda)_T$  strategy achieves moderate improvements over its elitist counterpart  $(\mu+\lambda)_E$  in some configurations, especially for lower values of  $\mu$  and  $\lambda$ . For instance, in blob-8 with  $\mu = 5, \lambda = 1$ ,  $(\mu+\lambda)_T$  yields an entropy of 1.199 (0.486), outperforming the corresponding elitist version (1.751 (0.378)) (see Table 1). However,

as population and offspring sizes increase, both  $(\mu+\lambda)$  variants tend to converge to similar or worse entropy values, with wider dispersion, suggesting that neither approach reliably maintains diversity without a full replacement scheme.

MG-MD GAN achieves reasonably low entropy values on blob-8, median (iqr) results are: 1.950 (0.062), 1.924 (0.123), and 1.920 (0.073) for population sizes 3, 5, and 7, respectively. These results are close to the  $(\mu+\lambda)_E$  and  $(\mu+\lambda)_T$  ones in some settings. However, on gaussian-4, all entropy values collapse to 0.0 regardless of population size, clearly indicating mode collapse and confirming that some evolutionary dynamics are necessary to preserve diversity in structured distributions.

**Table 1: Cluster-based entropy median (iqr) for  $(\mu + \lambda)_E$  and  $(\mu + \lambda)_T$  on blob-8 and gaussian-4. The lower the better.**

$\mu$	$\lambda$	blob-8		gaussian-4	
		$(\mu + \lambda)_E$	$(\mu + \lambda)_T$	$(\mu + \lambda)_E$	$(\mu + \lambda)_T$
3	1	1.513 (1.290)	1.257 (0.606)	0.693 (1.384)	0.582 (0.692)
3	2	1.675 (0.914)	1.946 (1.105)	0.693 (1.383)	0.691 (1.384)
3	3	1.917 (0.401)	1.992 (0.035)	0.693 (0.693)	0.692 (1.365)
5	1	1.751 (0.378)	1.199 (0.486)	0.693 (0.905)	0.429 (0.682)
5	3	1.843 (0.321)	1.943 (0.662)	0.693 (0.208)	0.344 (1.378)
5	5	1.572 (0.849)	1.956 (0.194)	0.692 (0.693)	0.346 (1.384)
7	1	1.630 (0.272)	1.131 (0.328)	0.692 (0.258)	0.512 (0.692)
7	4	1.838 (0.732)	1.803 (1.296)	0.693 (1.122)	0.692 (0.693)
7	7	1.909 (0.168)	1.917 (0.659)	0.693 (1.385)	0.000 (0.773)

**Table 2: Cluster-based entropy for  $(\mu, \lambda)_E$  on blob-8 and gaussian-4 in terms of median (iqr). The lower the better.**

$\mu$	$\lambda$	blob-8	gaussian-4
3	3	2.018 (0.130)	1.235 (1.384)
3	5	1.966 (0.115)	1.383 (0.148)
3	6	2.012 (0.070)	1.346 (0.417)
5	5	2.003 (0.086)	1.242 (0.378)
5	8	1.999 (0.075)	1.375 (0.867)
5	10	1.973 (0.049)	1.322 (1.384)
7	7	1.970 (0.047)	1.383 (0.324)
7	11	1.979 (0.073)	1.378 (1.039)
7	14	1.981 (0.052)	1.376 (0.644)

**Table 3: Mode coverage median (iqr) for  $(\mu + \lambda)_E$  and  $(\mu + \lambda)_T$  on blob-8 (ideal value 8) and gaussian-4 (ideal value 4).**

$\mu$	$\lambda$	blob-8		gaussian-4	
		$(\mu + \lambda)_E$	$(\mu + \lambda)_T$	$(\mu + \lambda)_E$	$(\mu + \lambda)_T$
3	1	5.5 (6.0)	4.0 (2.25)	2.0 (3.00)	2.0 (1.00)
3	2	5.5 (6.0)	8.0 (6.00)	2.0 (2.25)	2.0 (3.00)
3	3	8.0 (4.0)	8.0 (0.00)	2.0 (1.00)	2.0 (3.00)
5	1	6.0 (4.0)	4.0 (1.25)	2.0 (1.25)	2.0 (1.00)
5	3	8.0 (3.5)	8.0 (5.25)	2.0 (0.25)	1.5 (3.00)
5	5	5.0 (5.5)	8.0 (1.00)	2.0 (1.00)	1.5 (3.00)
7	1	5.0 (1.5)	3.0 (1.50)	2.0 (2.00)	2.0 (0.50)
7	4	7.0 (4.5)	8.0 (6.00)	2.0 (2.25)	2.0 (1.00)
7	7	8.0 (1.0)	8.0 (5.00)	2.0 (3.00)	1.0 (1.25)

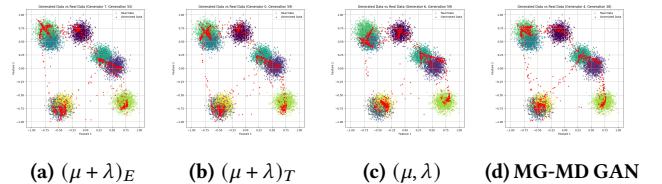
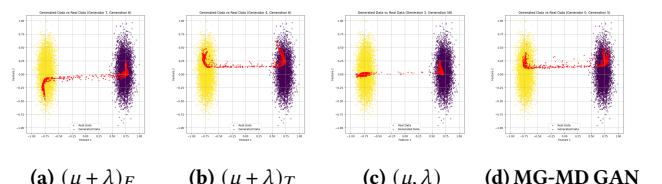
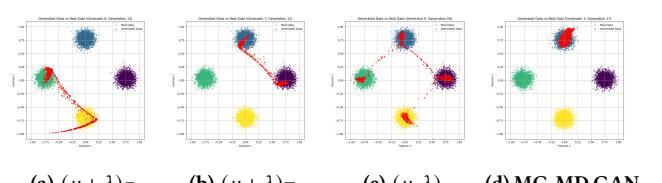
**Table 4: Mode coverage median (iqr) for  $(\mu, \lambda)$  on blob-8 (ideal value 8) and gaussian-4 (ideal value 4).**

$\mu$	$\lambda$	blob-8	gaussian-4
3	3	8.0 (0.0)	3.5 (2.0)
3	5	8.0 (0.0)	4.0 (0.25)
3	6	8.0 (0.0)	4.0 (1.00)
5	5	8.0 (0.0)	4.0 (1.25)
5	8	8.0 (0.0)	4.0 (2.25)
5	10	8.0 (0.0)	4.0 (3.00)
7	7	8.0 (0.0)	4.0 (0.25)
7	11	8.0 (0.0)	4.0 (2.50)
7	14	8.0 (0.0)	4.0 (1.00)

Mode coverage analysis reinforces these findings. As shown in Table 4,  $(\mu, \lambda)$  achieves perfect mode coverage (median 8.0 on blob-8, 4.0 on gaussian-4) in all evaluated configurations, with negligible or zero variance. This indicates not only robust diversity but also full mode representation across all runs. In contrast, the  $(\mu + \lambda)$  variants, especially  $(\mu + \lambda)_E$ , show frequent mode under-coverage and high variability. For instance, on gaussian-4 with  $\mu = 7$ ,  $\lambda = 7$ ,  $(\mu + \lambda)_E$  achieves a median mode count of only 2.0, while  $(\mu + \lambda)_T$  drops further to 1.0 (Table 3).

MG-MD GAN reaches full mode coverage (8.0) on blob-8 for all population sizes. However, it fails completely on gaussian-4, with a consistent median of 1.0. This further emphasizes that although static diversity can be sufficient in unstructured datasets, more targeted evolutionary mechanisms are required to handle complex data distributions with multiple tightly clustered modes.

Figures 3, 4, 5, and 6 present a representative set of the synthetic samples created by each one of the methods for blob-8, gaussian-2, gaussian-4, and gaussian-8 datasets, respectively. In these figures, red dots represent the produced samples.

**Figure 3: blob-8 samples produced when using  $\mu=3$ .****Figure 4: gaussian-2 samples produced when using  $\mu=3$ .****Figure 5: gaussian-4 samples produced when using  $\mu=3$ .**

A general trend can be seen in the visual results in figures 3-6 in which  $(\mu, \lambda)$  GAN produced samples closer to the real dataset with fewer samples out of the modes that define the datasets than the other approaches.

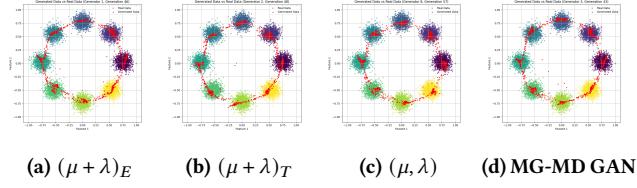
Focusing on the most challenging dataset, i.e., gaussian-4, the differences between the samples produced by  $(\mu, \lambda)$  method is remarkable, since it was capable to produce samples for all the four

**Table 5: Median FID (iqr) for each experiment variant by  $\mu$  and  $\lambda$ .**

$\mu$	$(\mu + \lambda)_E \lambda=1$	$(\mu + \lambda)_E \lambda=\mu$	$(\mu + \lambda)_T \lambda=1$	$(\mu + \lambda)_T \lambda=\mu$	$(\mu, \lambda) \lambda=\mu$	$(\mu, \lambda) \lambda=2\mu$	MG-MD GAN
3	55.463 (1.645)	52.321 (2.588)	53.488 (3.282)	50.178 (1.715)	47.156 (0.668)	45.218 (0.825)	48.357 (1.551)
5	52.760 (1.309)	50.221 (1.888)	52.030 (2.234)	48.475 (1.657)	46.145 (1.063)	44.074 (1.034)	46.622 (1.678)

**Table 6: Median TVD (iqr) for each experiment variant by  $\mu$  and  $\lambda$ .**

$\mu$	$(\mu + \lambda)_E \lambda=1$	$(\mu + \lambda)_E \lambda=\mu$	$(\mu + \lambda)_T \lambda=1$	$(\mu + \lambda)_T \lambda=\mu$	$(\mu, \lambda) \lambda=\mu$	$(\mu, \lambda) \lambda=2\mu$	MG-MD GAN
3	1.975 (0.383)	1.822 (0.221)	1.901 (0.450)	1.689 (0.255)	1.633 (0.516)	1.518 (0.362)	1.646 (0.263)
5	1.943 (0.224)	1.743 (0.238)	1.636 (0.386)	1.590 (0.433)	1.475 (0.320)	1.467 (0.206)	1.603 (0.284)

**Figure 6: gaussian-8 samples produced when using  $\mu=3$ .**

modes (see Figure 5). In contrast,  $(\mu + \lambda)$  variations collapse in two modes and MG-MD GAN collapses in only one mode.

Taken together, these results demonstrate that while non-evolutionary methods can yield competitive quality and mode coverage on simpler datasets, coevolutionary approaches with full generational replacement (i.e.,  $(\mu, \lambda)$ ) provide the most consistent and reliable strategy for maintaining diversity and avoiding mode collapse across a wide range of configurations and dataset complexities.

## 5.2 Results on MNIST

To limit computational cost in MNIST, we focus on the most promising configurations identified in the previous: population sizes  $\mu \in \{3, 5\}$  and offspring sizes  $\lambda \in \{1, \mu\}$  for  $(\mu+\lambda)$  and  $\lambda \in \{\mu, 2\mu\}$  for  $(\mu, \lambda)$ . These settings balance exploration and exploitation while remaining computationally feasible. The resulting FID and TVD scores are summarized numerically in table 5 and 6, which reports the median and interquartile range (iqr) for each configuration.

The results reveal several notable trends. First, the  $(\mu, \lambda)$  method, which implements a coevolutionary GAN training strategy based on the  $(\mu, \lambda)$  evolutionary algorithm without elitism, consistently achieves the lowest FID scores, indicating the best generative performance overall. Across both  $\mu = 3$  and  $\mu = 5$ , it shows strong medians and very narrow IQRs, suggesting not only high-quality outputs but also robust and stable behavior across runs.

In contrast, the  $(\mu + \lambda)_E$  method, which applies pure elitism, while occasionally producing competitive results, generally yields higher FID scores, particularly for smaller values of  $\lambda$ . Although elitism is often considered beneficial for preserving top individuals in evolutionary computation, these results suggest that in the context of coevolutionary GAN training, it may introduce premature convergence or limit exploration, especially in the early training

stages. The  $(\mu + \lambda)_T$  variant, using tournament selection, performs similarly to the elitist variant but with greater variability.

The MG-MD GAN approach, which trains populations of generators and discriminators without using evolutionary algorithms, performs reasonably well in some configurations but does not match the performance of the  $(\mu, \lambda)$  method. This further supports the benefit of explicitly applying evolutionary search dynamics in co-evolutionary GAN training.

A pairwise Wilcoxon signed-rank test with Bonferroni correction was conducted to assess statistical significance. The results confirm that  $(\mu, \lambda)$  is the most competitive method, with significantly better FID scores than most other configurations. In particular, the variant with  $\mu = 5, \lambda = 10$  achieved the lowest overall median. Conversely, the  $(\mu + \lambda)_E$  approach showed statistically significant worse performance in several pairwise comparisons, especially when  $\lambda$  was small.

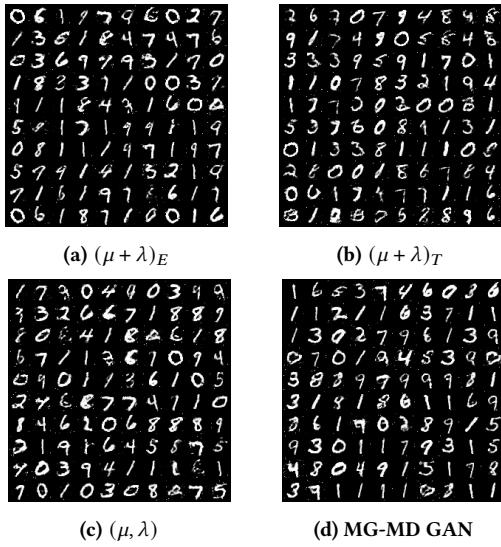
In terms of diversity, the TVD results in Table 6 and the Wilcoxon signed-rank test with Bonferroni correction show that the  $(\mu, \lambda)$  outperforms the other methods. These variants got the lowest TVD scores overall, indicating that the distribution of generated digits closely matches the real data label distribution. Specifically, the  $(\mu=5, \lambda=10)$  configuration achieves the best diversity result with a median TVD of 1.467, followed closely by  $(\mu=5, \lambda=5)$ ,  $(\mu=3, \lambda=6)$ , and  $(\mu=3, \lambda=3)$ , confirming a clear advantage for full generational replacement strategies.

The  $(\mu+\lambda)$  variations generally obtained higher TVD scores and wider interquartile ranges, reflecting less consistent alignment between the generated and real data distributions. Using tournament selection outperforms the elitist approach. This improvement can be attributed to the greater exploration enabled by tournament selection, which fosters population diversity and reduces the risk of premature convergence. Although some  $(\mu+\lambda)_T$  configurations approach the performance of  $(\mu, \lambda)$  in specific settings, e.g.,  $(\mu=5, \lambda=\mu)$ , their overall results remain less robust. The MG-MD GAN baseline also performs competitively in some cases but falls short of the best  $(\mu, \lambda)$  variants, reinforcing the importance of evolutionary dynamics for maintaining class-level diversity in image generation tasks.

Besides, all evaluated methods achieved the ideal mode coverage of 10, indicating that each digit class was represented in the generated samples. However, the TVD results highlight that merely covering all classes is not sufficient, the generated distributions must also approximate the real data distribution closely. As it has

been discussed in the blob and gaussian results, these findings underline the critical role of population diversity and controlled evolutionary pressure in avoiding mode imbalance and ensuring stable, high-quality generation across all output classes.

Figure 7 shows some samples produced by the analyzed approaches when using population size 3. It can be observed that in general all the approaches produced high quality samples. However,  $(\mu + \lambda)_E$  samples set has lower diversity, i.e., high frequency of one digit, than the set produced by  $(\mu, \lambda)$ .



**Figure 7: MNIST samples produced when using  $\mu=3$ .**

### 5.3 General Discussion

This section discussed the main findings in the context of the research questions introduced in Section 1.

**RQ1:** How do different coevolutionary strategies—particularly selection and replacement mechanisms—impact the quality and diversity of GAN-generated samples? The experiments show that the choice of selection and replacement strategy has a major effect on both quality and diversity. Full generational replacement strategies, i.e.,  $(\mu, \lambda)$ , consistently outperform elitist and tournament-based  $(\mu+\lambda)$ . Elitism promotes exploitation but often leads to premature convergence and mode collapse. In contrast, tournament selection combined with generational replacement fosters diversity and leads to better and more stable outcomes.

**RQ2:** To what extent does the balance between exploration and exploitation, as controlled by population and offspring sizes, affect the performance and stability of GAN training? Increasing offspring size generally improves performance by enhancing exploratory capacity, particularly in tournament-based methods. However, large populations alone are not sufficient. The best results are obtained when larger offspring sizes are combined with full replacement, allowing broader exploration while avoiding dominance by early high performers. These effects are especially evident in structured datasets like gaussian-4 and MNIST.

**RQ3:** Can a simple population based multi-generator multi-discriminator (MG-MD) GAN without selection or replacement be competitive with coevolutionary approaches? MG-MD GAN performs competitively on simpler datasets, e.g., blob-8, and achieves full mode coverage. However, it fails to generalize to more structured or complex distributions, such as gaussian-4 and MNIST, where it suffers from mode collapse and reduced diversity. These results suggest that while static diversity can help in simple domains, evolutionary dynamics are necessary to achieve robustness and generalization in more challenging scenarios.

## 6 Conclusions and Future Work

This paper presents an empirical analysis of coevolutionary strategies for GAN training. We studied the impact of selection and replacement mechanisms across multiple configurations, comparing full generational replacement, i.e.,  $(\mu, \lambda)$ , and elitist and tournament-based  $(\mu+\lambda)$  approaches, along with a non-evolutionary MG-MD baseline. The evaluation was conducted on both synthetic 2D datasets and the MNIST image dataset.

The results consistently show that  $(\mu, \lambda)$  based variations outperform other variants in terms of both sample quality and diversity. These configurations encourage exploration by discarding parents and replacing them entirely with offspring, which prevents premature convergence and promotes robustness. Tournament-based selection improves over elitism, but only when coupled with sufficient exploration via offspring replacement. While the MG-MD baseline performs well on simple datasets, it fails to generalize to more structured or high-dimensional data, highlighting the importance of applying evolutionary pressure.

Future work will explore more complex datasets (e.g., CIFAR-10, SVHN), other coevolutionary mechanisms such as spatially structured populations or quality-diversity methods, and multi-objective formulations that explicitly balance quality and diversity. Additionally, we aim to extend this framework to assess coevolutionary dynamics in conditional and multi-modal generative models.

## References

- [1] Mohd Ali, Mehbob Ali, Mubashir Hussain, and Deepika Koundal. 2024. Generative adversarial networks (gans) for medical image processing: Recent advancements. *Archives of Computational Methods in Engineering* (2024), 1–14.
- [2] Luca Ambrogioni, Tue Herlau, and Mathias Bastian. 2021. Automatic evaluation of generative models using density-ratio estimation. *Neural Computation* 33, 5 (2021), 1119–1140.
- [3] L. M. Antonio and C. A. C. Coello. 2018. Coevolutionary Multi-objective Evolutionary Algorithms: A Survey of the State-of-the-Art. *IEEE Transactions on Evolutionary Computation* (2018), 1–16. doi:10.1109/TEVC.2017.2767023
- [4] Andrea Arcuri and Xin Yao. 2007. Coevolving programs and unit tests from their specification. In *Proceedings of the 22nd IEEE/ACM International Conference on Automated Software Engineering*, 397–400.
- [5] Andrea Arcuri and Xin Yao. 2014. Co-evolutionary automatic programming for software development. *Information Sciences* 259 (2014), 412–432.
- [6] Martin Arjovsky, Soumith Chintala, and Léon Bottou. 2017. Wasserstein generative adversarial networks. In *International conference on machine learning*. PMLR, 214–223.
- [7] Sanjeev Arora, Andrej Risteski, and Yi Zhang. 2018. Do GANs learn the distribution? some theory and empirics. In *International conference on learning representations*.
- [8] Robert Axelrod and William D Hamilton. 1981. The evolution of cooperation. *science* 211, 4489 (1981), 1390–1396.
- [9] Jing Bai, Yang Zhang, Zhu Xiao, Fawang Ye, You Li, Mamoun Alazab, and Licheng Jiao. 2022. Immune evolutionary generative adversarial networks for hyperspectral image classification. *IEEE Transactions on Geoscience and Remote Sensing* 60 (2022), 1–14.

- [10] Marco Baoletti, Carlos Artemio Coello Coello, Gabriele Di Bari, and Valentina Poggioni. 2020. Multi-objective evolutionary GAN. In *Proceedings of the 2020 genetic and evolutionary computation conference companion*. 1824–1831.
- [11] Earl T Barr, Mark Harman, Phil McMinn, Muzammil Shahbaz, and Shin Yoo. 2014. The oracle problem in software testing: A survey. *IEEE transactions on software engineering* 41, 5 (2014), 507–525.
- [12] Haripriya Chakraborty, Priyanka Samanta, and Liang Zhao. 2021. Sequential data imputation with evolving generative adversarial networks. In *2021 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 1–8.
- [13] Victor Costa, Nuno Lourenço, João Correia, and Penousal Machado. 2019. CO-EGAN: evaluating the coevolution effect in generative adversarial networks. In *Proceedings of the genetic and evolutionary computation conference*. 374–382.
- [14] Victor Costa, Nuno Lourenço, João Correia, and Penousal Machado. 2020. Neuroevolution of generative adversarial networks. *Deep Neural Evolution: Deep Learning with Evolutionary Computation* (2020), 293–322.
- [15] Victor Costa, Nuno Lourenço, João Correia, and Penousal Machado. 2021. Demonstrating the Evolution of GANs through t-SNE. In *Applications of Evolutionary Computation: 24th International Conference, EvoApplications 2021, Held as Part of EvoStar 2021, Virtual Event, April 7–9, 2021, Proceedings 24*. Springer, 618–633.
- [16] Victor Costa, Nuno Lourenço, and Penousal Machado. 2019. Coevolution of generative adversarial networks. In *Applications of Evolutionary Computation: 22nd International Conference, EvoApplications 2019, Held as Part of EvoStar 2019, Leipzig, Germany, April 24–26, 2019, Proceedings 22*. Springer, 473–487.
- [17] Li Deng. 2012. The mnist database of handwritten digit images for machine learning research [best of the web]. *IEEE signal processing magazine* 29, 6 (2012), 141–142.
- [18] Diana Flores, Erik Hemberg, Jamal Toutouh, and Una-May O'Reilly. 2022. Co-evolutionary generative adversarial networks for medical image augmentation at scale. In *Proceedings of the genetic and evolutionary computation conference*. 367–376.
- [19] Ying Fu, Minxue Gong, Guang Yang, Hong Wei, and Jiliu Zhou. 2021. Evolutionary gan-based data augmentation for cardiac magnetic resonance image. *Computers, Materials & Continua* 68, 1 (2021), 1359–1374.
- [20] Unai Garciairena, Roberto Santana, and Alexander Mendiburu. 2018. Evolved GANs for generating Pareto set approximations. In *Proceedings of the genetic and evolutionary computation conference*. 434–441.
- [21] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2020. Generative adversarial networks. *Commun. ACM* 63, 11 (2020), 139–144.
- [22] Juanjuan He, Qi Zhu, Kai Zhang, Piaoyao Yu, and Jinshan Tang. 2021. An evolvable adversarial network with gradient penalty for COVID-19 infection segmentation. *Applied Soft Computing* 113 (2021), 107947.
- [23] Erik Hemberg, Jamal Toutouh, Abdullah Al-Dujaili, Tom Schmiedlechner, and Una-May O'Reilly. 2021. Spatial coevolution for generative adversarial network training. *ACM Transactions on Evolutionary Learning and Optimization* 1, 2 (2021), 1–28.
- [24] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. 2017. GANs trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in neural information processing systems* 30 (2017).
- [25] Philip Hingston and Mike Preuss. 2011. Red teaming with coevolution. In *2011 IEEE Congress of Evolutionary Computation (CEC)*. IEEE, 1155–1163.
- [26] David Keaveney and Colm O'Riordan. 2011. Evolving coordination for real-time strategy games. *IEEE Transactions on Computational Intelligence and AI in Games* 3, 2 (2011), 155–167.
- [27] Krzysztof Krawiec and Malcolm Heywood. 2016. Solving Complex Problems with Coevolutionary Algorithms. In *Proceedings of the 2016 on Genetic and Evolutionary Computation Conference Companion*. ACM, 687–713.
- [28] Hyeun Ku and Minhyeok Lee. 2023. TextControlGAN: Text-to-image synthesis with controllable generative adversarial networks. *Applied Sciences* 13, 8 (2023), 5098.
- [29] Jerry Li, Aleksander Madry, John Peebles, and Ludwig Schmidt. 2018. On the limitations of first-order approximation in GAN dynamics. In *International Conference on Machine Learning*. PMLR, 3005–3013.
- [30] Luke Metz, Ben Poole, David Pfau, and Jascha Sohl-Dickstein. 2016. Unrolled generative adversarial networks. *arXiv preprint arXiv:1611.02163* (2016).
- [31] Jordan B Pollack and Alan D Blair. 1998. Co-evolution in the successful learning of backgammon strategy. *Machinel learning* 32 (1998), 225–240.
- [32] Elena Popovici, Anthony Bucci, R. Paul Wiegand, and Edwin D. De Jong. 2012. *Coevolutionary Principles*. Springer Berlin Heidelberg, Berlin, Heidelberg, 987–1033.
- [33] Christopher D Rosin and Richard K Belew. 1997. New methods for competitive coevolution. *Evolutionary Computation* 5, 1 (1997), 1–29.
- [34] George Rush, Daniel R Tauritz, and Alexander D Kent. 2015. Coevolutionary agent-based network defense lightweight event system (CANDLES). In *Proceedings of the Companion Publication of the 2015 Annual Conference on Genetic and Evolutionary Computation*. 859–866.
- [35] Francisco Sedeño, Jamal Toutouh, and Francisco Chicano. 2025. Generate more than one child in your co-evolutionary semi-supervised learning GAN. In *Proceedings of the The Leading European Event on Bio-Inspired AI (EVOSTAR)*. 16. doi:10.5281/zenodo.15124788
- [36] Travis Service and Daniel Tauritz. 2009. Increasing infrastructure resilience through competitive coevolution. *New Mathematics and Natural Computation* 5, 02 (2009), 441–457.
- [37] Karl Sims. 1994. Evolving 3D morphology and behavior by competition. *Artificial life* 1, 4 (1994), 353–372.
- [38] Moshe Sipper. 2011. *Evolved to Win*. Lulu. com.
- [39] Fanglei Sun, Qian Tao, Jianqiao Hu, and Jieqiong Liu. 2021. Composite evolutionary gan for natural language generation with temper control. In *2021 7th International Conference on Computer and Communications (ICCC)*. IEEE, 1710–1714.
- [40] Jamal Toutouh, Mathias Esteban, and Sergio Nesmachnow. 2020. Parallel/distributed generative adversarial neural networks for data augmentation of COVID-19 training images. In *Latin American High Performance Computing Conference*. Springer, 162–177.
- [41] Jamal Toutouh, Erik Hemberg, and Una-May O'Reilly. 2019. Spatial evolutionary generative adversarial networks. In *Proceedings of the genetic and evolutionary computation conference*. 472–480.
- [42] Jamal Toutouh, Erik Hemberg, and Una-May O'Reilly. 2020. Analyzing the components of distributed coevolutionary gan training. In *Parallel Problem Solving from Nature—PPSN XVI: 16th International Conference, PPSN 2020, Leiden, The Netherlands, September 5–9, 2020, Proceedings, Part I 16*. Springer, 552–566.
- [43] Jamal Toutouh, Erik Hemberg, and Una-May O'Reilly. 2020. Data dieting in gan training. In *Deep Neural Evolution: Deep Learning with Evolutionary Computation*. Springer, 379–400.
- [44] Jamal Toutouh, Subhash Nalluru, Erik Hemberg, and Una-May O'Reilly. 2023. Semi-supervised generative adversarial networks with spatial coevolution for enhanced image generation and classification. *Applied Soft Computing* 148 (2023), 110890.
- [45] Jamal Toutouh and Una-May O'Reilly. 2021. Signal propagation in a gradient-based and evolutionary learning system. In *Proceedings of the Genetic and Evolutionary Computation Conference*. 377–385.
- [46] S.S. Vallender. 1974. Calculation of the Wasserstein distance between probability distributions on the line. *Theory of Probability & Its Applications* 18, 4 (1974), 784–786.
- [47] Chaoyue Wang, Chang Xu, Xin Yao, and Dacheng Tao. 2019. Evolutionary generative adversarial networks. *IEEE Transactions on Evolutionary Computation* 23, 6 (2019), 921–934.
- [48] Gabriel Wang, Anish Thite, Rodd Talebi, Anthony D'Achille, Alex Mussa, and Jason Zutty. 2022. Evolving SimGANs to improve abnormal electrocardiogram classification. In *Proceedings of the Genetic and Evolutionary Computation Conference Companion*. 1887–1894.
- [49] Josh L Wilkerson and Daniel Tauritz. 2010. Coevolutionary automated software correction. In *Proceedings of the 12th annual conference on Genetic and evolutionary computation*. 1391–1392.
- [50] Guohao Ying, Xin He, Bin Gao, Bo Han, and Xiaowen Chu. 2022. EAGAN: Efficient two-stage evolutionary architecture search for GANs. In *European conference on computer vision*. Springer, 37–53.
- [51] Hongke Zhao, Xinpeng Wu, Chuang Zhao, Lei Zhang, Haiping Ma, and Fan Cheng. 2021. Coea: A cooperative–competitive evolutionary algorithm for bidirectional recommendations. *IEEE Transactions on Evolutionary Computation* 26, 1 (2021), 28–42.
- [52] Wenbo Zheng, Chao Gou, Lan Yan, and Fei-Yue Wang. 2019. Differential-evolution-based generative adversarial networks for edge detection. In *Proceedings of the IEEE/CVF international conference on computer vision workshops*. 0–0.