Evolutionary Generative Adversarial Networks

Chaoyue Wang, Chang Xu, Xin Yao, *Fellow, IEEE*, and Dacheng Tao, *Fellow, IEEE*

***Abstract*—Generative adversarial networks (GANs) have been effective for learning generative models for real-world data. However, accompanied with the generative tasks becoming more and more challenging, existing GANs (GAN and its variants) tend to suffer from different training problems such as instability and mode collapse. In this paper, we propose a novel GAN framework called evolutionary GANs (E-GANs) for stable GAN training and improved generative performance. Unlike existing GANs, which employ a predefined adversarial objective function alternately training a generator and a discriminator, we evolve a population of generators to play the adversarial game with the discriminator. Different adversarial training objectives are employed as mutation operations and each individual (i.e., generator candidature) are updated based on these mutations. Then, we devise an evaluation mechanism to measure the quality and diversity of generated samples, such that only well-performing generator(s) are preserved and used for further training. In this way, E-GAN overcomes the limitations of an individual adversarial training objective and always preserves the well-performing offspring, contributing to progress in, and the success of GANs. Experiments on several datasets demonstrate that E-GAN achieves convincing generative performance and reduces the training problems inherent in existing GANs.**

***Index Terms*—Deep generative models, evolutionary computation, generative adversarial networks (GANs).**

I. Introduction

GENERATIVE adversarial networks (GANs) [1] are one of the main groups of methods used to learn generative models from complicated real-world data. As well as using a generator to synthesize semantically meaningful data from standard signal distributions, GANs (GAN and its variants) train a discriminator to distinguish *real* samples in the training dataset from *fake* samples synthesized by the generator. As the confronter, the generator aims to deceive the discriminator by producing ever more realistic samples. The training procedure continues until the generator wins the adversarial game; that is, the discriminator cannot make a better decision than randomly guessing whether a particular sample is fake or real. Compared with other existing generative models, GANs provide a concise and efficient framework for learning generative models. Therefore, GANs have recently been successfully applied to image generation [2]–[6], image editing [7]–[10], video prediction [11]–[13], and many other tasks [14]–[18].

Comparing with most existing discriminative tasks (e.g., classification and clustering), GANs perform a challenging generative process, which projects a standard distribution to a much more complex high-dimensional real-world data distribution. First, although GANs have been utilized to model large-scale real-world datasets, such as CelebA, LSUN, and ImageNet, they easily suffer from the mode collapse problem, i.e., the generator can only learn some limited patterns from the large-scale target datasets, or assigns all of its probability mass to a small region in the space [19]. Meanwhile, if the generated distribution and the target data distribution do not substantially overlap (usually at the beginning of training), the generator gradients can point to more or less random directions or even result in the vanishing gradient issue. In addition, to generate high-resolution and high-quality samples, both the generator and discriminator are asked to be deeper and larger. Under the vulnerable adversarial framework, it is hard to balance and optimize such large-scale deep networks. Thus, in most of the existing works, appropriate hyper-parameters (e.g., learning rate and updating steps) and network architectures are critical configurations. Unsuitable settings reduce GAN's performance or even fail to produce any reasonable results. Overall, although GANs already produce visually appealing samples in various applications, they are still facing many large-scale optimization problems.

Many recent efforts on GANs have focused on overcoming these optimization difficulties by developing various adversarial training objectives. Typically, assuming the optimal discriminator for the given generator is learned, different objective functions of the generator aim to measure the distance between the generated distribution and the target data distribution under different metrics. The original GAN uses Jensen–Shannon divergence (JSD) as the metric. A number of metrics have been introduced to improve GAN's performance, such as least squares [20], absolute

Manuscript received August 16, 2018; revised November 29, 2018; accepted January 15, 2019. Date of publication January 28, 2019; date of current version November 27, 2019. This work was supported in part by the Australian Research Council under Project FL-170100117, Project DP-180103424, Project IH180100002, and Project DE180101438, in part by the National Key Research and Development Program of China under Grant 2017YFC0804003, in part by the Engineering and Physical Sciences Research Council (EPSRC) under Grant EP/J017515/1 and Grant EP/P005578/1, in part by the Program for Guangdong Introducing Innovative and Enterpreneurial Teams under Grant 2017ZT07X386, in part by the Shenzhen Peacock Plan under Grant KQTD2016112514355531, in part by the Science and Technology Innovation Committee Foundation of Shenzhen under Grant ZDSYS201703031748284, and in part by the Program for University Key Laboratory of Guangdong Province under Grant 2017KSYS008. *(Corresponding author: Chang Xu.)*

C. Wang, C. Xu, and D. Tao are with the UBTECH Sydney Artificial Intelligence Centre, Faculty of Engineering and Information Technologies, University of Sydney, Darlington, NSW 2008, Australia, and also with the School of Computer Science, University of Sydney, Darlington, NSW 2008, Australia (e-mail: chaoyue.wang@sydney.edu.au; c.xu@sydney.edu.au; dacheng.tao@sydney.edu.au).

X. Yao is with the Shenzhen Key Laboratory of Computational Intelligence, University Key Laboratory of Evolving Intelligent Systems of Guangdong Province, Department of Computer Science and Engineering, Southern University of Science and Technology, Shenzhen 518055, China, and also with the CERCIA, School of Computer Science, University of Birmingham, Birmingham B15 2TT, U.K. (e-mail: x.yao@cs.bham.ac.uk).

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

Digital Object Identifier 10.1109/TEVC.2019.2895748

deviation [21], Kullback–Leibler (KL) divergence [22], and Wasserstein distance [23]. However, according to both theoretical analyses and experimental results, minimizing each distance has its own pros and cons. For example, although measuring KL divergence largely eliminates the vanishing gradient issue, it easily results in mode collapse [22], [24]. Likewise, Wasserstein distance greatly improves training stability but can have nonconvergent limit cycles near equilibrium [25].

Through observation, we find that most existing GAN methods are limited by the specified adversarial optimization strategy. Since the training strategy is fixed, it is hard to adjust the balance between the generator and discriminator during the training process. Meanwhile, as aforementioned, each existing adversarial training strategy has its own pros and cons in training GAN models.

In this paper, we build an evolutionary GAN (E-GAN), which treats the adversarial training procedure as an evolutionary problem. Specifically, a discriminator acts as the *environment* (i.e., provides adaptive loss functions) and a *population* of generators evolve in response to the environment. During each adversarial (or evolutionary) iteration, the discriminator is still trained to recognize real and fake samples. However, in our method, acting as parents, generators undergo different *mutations* to produce offspring to adapt to the environment. Different adversarial objective functions aim to minimize different distances between the generated distribution and the data distribution, leading to the different mutations. Meanwhile, given the current optimal discriminator, we measure the quality and diversity of samples generated by the updated offspring. Finally, according to the principle of "survival of the fittest," poorly performing offspring are removed and the remaining well-performing offspring (i.e., generators) are preserved and used for further training. Based on the evolutionary paradigm to optimize GANs, the proposed E-GAN overcomes the inherent limitations in the individual adversarial training objectives and always preserves the well-performing offspring produced by different training objectives (i.e., mutations). In this way, we contribute to progress in and the success of the large-scale optimization of GANs. Following vanilla GAN [1], we evaluate the new algorithm in image generation tasks. Overall, the proposed evolutionary strategy is largely orthogonal to the existing GAN models. Through applying the evolutionary framework to different GAN models, it is possible to perform different kinds of generation tasks. For example, GAN objectives were devised for generating text. Considering them as the mutation operations, the proposed evolutionary framework can be applied to solve text generation tasks.

Meanwhile, the proposed E-GAN framework also provides a novel direction to apply evolutionary learning paradigm on solving deep learning problems. Recent years, although deep learning algorithms have achieved promising performances on a variety of applications, they are still facing many challenges in solving real-world problems. Evolutionary computation, as a powerful approach to complex real-world problems [26]–[29], has been utilized to solve many deep learning challenges. Among them, Real *et al*. [30] devised an evolutionary algorithm to automatically search the architecture and hyper-parameters of deep networks. Moreover, evolution strategies have been utilized as an alternative to MDP-based techniques to optimize reinforcement learning models [31]. In this paper, we attempt to combine the back propagation algorithm and the evolutionary algorithm for optimizing deep generative models. The parameters updated by different learning objectives are regarded as variation results during the evolutionary process. By introducing suitable evaluation and selection mechanisms, the whole training process can be more efficient and stable. We hope the proposed evolutionary learning framework can be generalized to more deep learning problems, such as reinforcement learning.

In summary, we make the following contributions in this paper.

1) In order to stabilize GAN's training process, we devised a simple yet efficient evolutionary algorithm for optimizing generators within GANs framework. To the best of our knowledge, it is the first work that introduces the evolutionary learning paradigm into learning GAN models.
2) Through analyzing the training process of E-GAN, some properties of existing GANs objectives are further explored and discussed.
3) Experiments evaluated on several large-scale datasets are performed, and demonstrate that convincing results can be achieved by the proposed E-GAN framework.

The rest of this paper is organized as follows. After a brief summary of the previous related works in Section II, we illustrate the proposed E-GAN together with its training process in Section III. Then, we exhibit the experimental validation of the whole method in Section IV. Finally, we conclude this paper with some future directions in Section V.

II. Background

In this section, we first review some previous GANs devoted to reducing training instability and improving the generative performance. We then briefly summarize some of the evolutionary algorithms on deep neural networks.

A. Generative Adversarial Networks

GANs provides an excellent framework for learning deep generative models, which aim to capture the probability distributions over the given data. Compared to other generative models, GAN is easily trained by alternately updating a generator and a discriminator using the back propagation algorithm. In many generative tasks, GANs (GAN and its variants) produce better samples than other generative models [32]. Besides image generation tasks, GANs have been introduced to more and more tasks, such as video generation [11], [33], visual tracking [34]–[36], domain adaption [37], hashing coding [38]–[40], and feature learning [41], [42]. In these tasks, the adversarial training strategy also achieved promising performances.

However, some problems still exist in the GANs training process. In the original GAN, training the generator was equal to minimizing the JSD between the data distribution and the generated distribution, which easily resulted in the vanishing gradient problem. To solve this issue, a nonsaturating heuristic

objective (i.e., "$-\log D$ trick") replaced the minimax objective function to penalize the generator [1]. Then, Radford *et al.* [22] and Salimans *et al.* [43] designed specified network (DCGAN) architectures and proposed several heuristic tricks (e.g., feature matching, one-side label smoothing, and virtual batch normalization) to improve training stability. Meanwhile, energy-based GAN [21] and least-squares GAN [20] improved training stability by employing different training objectives. Although these methods partly enhanced training stability, in practice, the network architectures, and training procedure still required careful design to maintain the discriminator-generator balance. More recently, Wasserstein GAN (WGAN) [23] and its variant WGAN-GP (with gradient penalty) [44] were proposed to minimize the Wasserstein-1 distance between the generated and data distributions. Since the Wasserstein-1 distance is continuous everywhere and differentiable almost everywhere under only minimal assumptions [23], these two methods convincingly reduce training instability. However, to measure the Wasserstein-1 distance between the generated distribution and the data distribution, they are asked to enforce the Lipschitz constraint on the discriminator (also known as critic), which may result in some optimization problems [44].

Besides devising different objective functions, some works attempt to stable GAN training and improve generative performance by introducing multiple generators or discriminators into an adversarial training framework. Nguyen *et al.* [45] proposed a dual discriminator GAN, which combines the KL and reverse KL divergences into a unified objective function through employing two discriminators. Multidiscriminator GAN frameworks [46], [47] are devised and utilized for providing stable gradients to the generator and further stabilizing the adversarial training process. Moreover, Tolstikhin *et al.* [48] applied boosting techniques to train a mixture of generators by continually adding new generators to the mixture. Ghosh *et al.* [49] trained many generators by using a multiclass discriminator that predicts which generator produces the sample. Mixture GAN [50] is proposed to overcome the mode collapse problem by training multiple generators to specialize in different data modes. Overall, these multigenerator GANs aim to learn a set of generators, and the mixture of their learned distributions would approximate the data distribution, i.e., different generators are encouraged to capture different data modes. Although the proposed E-GAN also creates multiple generators during training, we always keep the well-performing candidates through survival of the fittest, which helps the final learned generator achieving better performance. Note that, in our framework, only one generator was learned to represent the whole target distribution at the end of the training.

B. Evolutionary Computation

Over the last 20 years, evolutionary algorithms have achieved considerable success across a wide range of computational tasks, including modeling, optimization, and design [51]–[55]. Inspired by natural evolution, the essence of an evolutionary algorithm is to equate possible solutions to individuals in a population, produce offspring through variations, and select appropriate solutions according to fitness [56].

Recently, evolutionary algorithms have been introduced to solve deep learning problems. To minimize human participation in designing deep algorithms and automatically discover such configurations, there have been many attempts to optimize deep learning hyper-parameters and design deep network architectures through an evolutionary search [57], [58]. Among them, Real *et al.* [30] proposed a large-scale evolutionary algorithm to design a whole deep classifier automatically. Meanwhile, different from widely employed gradient-based learning algorithms (e.g., backpropagation), evolutionary algorithms have also demonstrated their capacity to optimize neural networks. EPNet [59] was devised for evolving and training neural networks using evolutionary programming. In [60], EvoAE was proposed to speed up the training of autoencoders for constructing deep neural networks. Moreover, Salimans *et al.* [31] proposed a novel evolutionary strategy as an alternative to the popular MDP-based reinforcement learning techniques, achieving strong performance on reinforcement learning benchmarks. In addition, an evolutionary algorithm was proposed to compress deep learning models by automatically eliminating redundant convolution filters [61]. Last but not the least, the evolutionary learning paradigm has been utilized to solve a number of deep/machine tasks, such as automatic machine learning [62], multiobjective optimization [63], etc. However, to the best of our knowledge, there is still no work attempt to optimize deep generative models with the evolutionary learning algorithms.

III. Methods

In this section, we first review the original GAN formulation. Then, we introduce the proposed E-GAN algorithm. By illustrating E-GAN's mutations and evaluation mechanism, we further discuss the advantage of the proposed framework. Finally, we conclude with the entire E-GAN training process.

A. Generative Adversarial Networks

GAN, first proposed in [1], studies a two-player minimax game between a discriminative network D and a generative network G. Taking noisy sample $z \sim p(z)$ (sampled from a uniform or normal distribution) as the input, the generative network G outputs new data $G(z)$, whose distribution p_g is supposed to be close to that of the data distribution p_{data}. Meanwhile, the discriminative network D is employed to distinguish the true data sample $x \sim p_{\text{data}}(x)$ and the generated sample $G(z) \sim p_g(G(z))$. In the original GAN, this adversarial training process was formulated as

$$\min_G \max_D \mathbb{E}_{x\sim p_{\text{data}}}\big[\log D(x)\big] + \mathbb{E}_{z\sim p_z}\big[\log(1 - D(G(z)))\big]. \quad (1)$$

The adversarial procedure is illustrated in Fig. 1(a). Most existing GANs perform a similar adversarial procedure in different adversarial objective functions.

B. Evolutionary Algorithm

In contrast to conventional GANs, which alternately update a generator and a discriminator, we devise an evolutionary algorithm that evolves a population of generator(s) $\{G\}$ in a

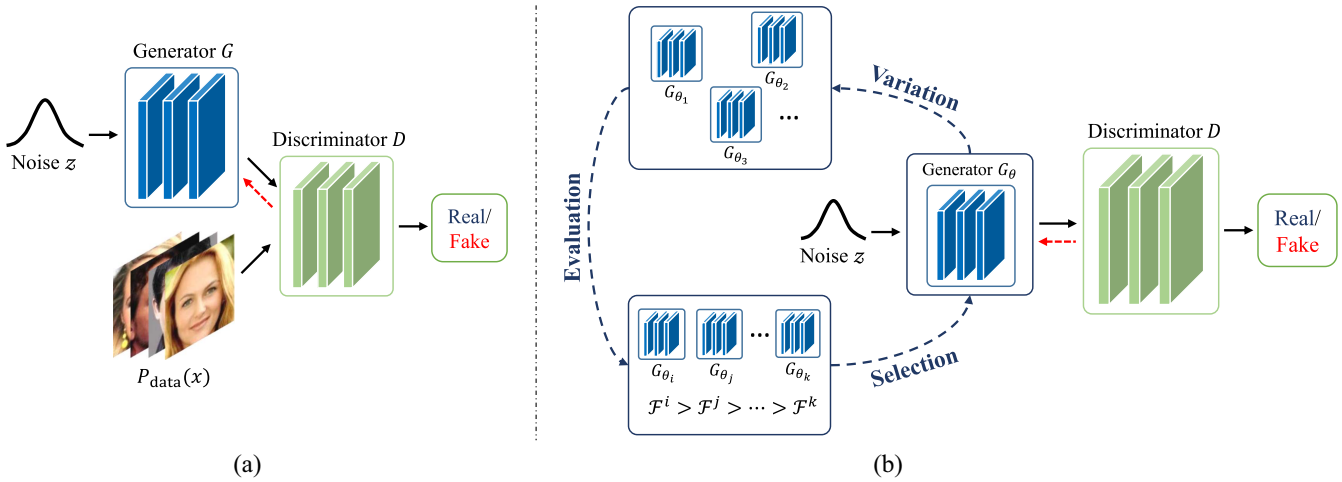

Fig. 1. (a) Original GAN framework. A generator G and a discriminator D play a two-player adversarial game. The updating gradients of the generator G are received from the adaptive objective, which depends on discriminator D. (b) Proposed E-GAN framework. A population of generators $\{G_\theta\}$ evolves in a dynamic environment, the discriminator D. Each evolutionary step consists of three substages: variation, evaluation, and selection. The best offspring are kept.

given environment (i.e., the discriminator D). In this population, each *individual* represents a possible solution in the parameter space of the generative network G. During the evolutionary process, we expect that the population gradually adapts to its environment, which means that the evolved generator(s) can generate ever more realistic samples and eventually learn the real-world data distribution. As shown in Fig. 1(b), during evolution, each step consists of three substages.

1) *Variation:* Given an individual G_θ in the population, we utilize the variation operators to produce its offspring $\{G_{\theta_1}, G_{\theta_2}, \ldots\}$. Specifically, several copies of each individual—or *parent*—are created, each of which are modified by different *mutations*. Then, each modified copy is regarded as one *child*.
2) *Evaluation:* For each child, its performance—or *individual's quality*—is evaluated by a *fitness* function $\mathcal{F}(\cdot)$ that depends on the current environment (i.e., discriminator D).
3) *Selection:* All children will be selected according to their fitness value, and the worst part is removed—that is, they are *killed*. The rest remain *alive* (i.e., free to act as parents) and evolve to the next iteration.

After each evolutionary step, the discriminative network D (i.e., the environment) is updated to further distinguish real samples x and fake samples y generated by the evolved generator(s), i.e.,

$$\mathcal{L}_D = -\mathbb{E}_{x\sim p_{\text{data}}}\big[\log D(x)\big] - \mathbb{E}_{y\sim p_g}\big[\log(1 - D(y))\big]. \quad (2)$$

Thus, the discriminative network D (i.e., the environment) can continually provide the adaptive losses to drive the population of generator(s) evolving to produce better solutions. Next, we illustrate and discuss the proposed variation (or mutation), evaluation, and selection operators in detail.

C. Variation

We employ *asexual reproduction* with different mutations to produce the next generation's individuals (i.e., children). Specifically, these mutation operators correspond to different training objectives, which attempt to narrow the distances between the generated distribution and the data distribution from different perspectives. In this section, we introduce the mutations used in this paper.[1] To analyze the corresponding properties of these mutations, we suppose that, for each evolutionary step, the optimal discriminator $D^*(x) = ([p_{\text{data}}(x)]/[p_{\text{data}}(x) + p_g(x)])$, according to (2), has already been learned [1].

1) Minimax Mutation: The minimax mutation corresponds to the minimax objective function in the original GAN

$$\mathcal{M}_G^{\text{minimax}} = \frac{1}{2}\mathbb{E}_{z\sim p_z}\big[\log(1 - D(G(z))\big]. \quad (3)$$

According to the theoretical analysis in [1], given the optimal discriminator D^*, the minimax mutation aims to minimize the JSD between the data distribution and the generated distribution. Although the minimax game is easy to explain and theoretically analyze, its performance in practice is disappointing, a primary problem being the generator's vanishing gradient. If the support of two distributions lies in two manifolds, the JSD will be a constant, leading to the vanishing gradient [24]. This problem is also illustrated in Fig. 2. When the discriminator rejects generated samples with high confidence [i.e., $D(G(z)) \to 0$], the gradient tends to vanishing. However, if the generated distribution overlaps with the data distribution, meaning that the discriminator cannot completely distinguish real from fake samples, the minimax mutation provides effective gradients and continually narrows the gap between the data distribution and the generated distribution.

2) Heuristic Mutation: Unlike the minimax mutation, which minimizes the log probability of the discriminator being correct, the heuristic mutation aims to maximize the log

[1] Although more mutation operations can be included in our framework, according to the theoretical analysis below, we adopt three interpretable and complementary objectives as our mutations. Meanwhile, we have tested more mutation operations, yet the mutations described in this paper already delivered a convincing performance.

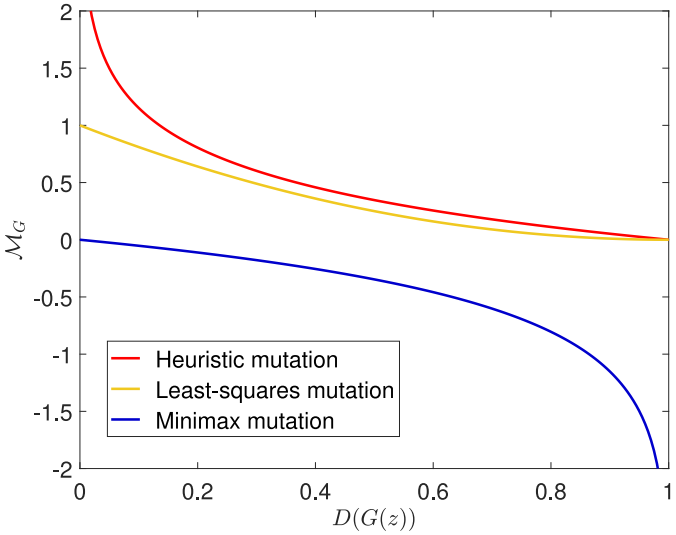

Fig. 2. Mutation (or objective) functions that the generator G receives given the discriminator D.

probability of the discriminator being mistaken, i.e.,

$$\mathcal{M}_G^{\text{heuristic}} = -\frac{1}{2}\mathbb{E}_{z\sim p_z}\big[\log(D(G(z))\big]. \tag{4}$$

Compared to the minimax mutation, the heuristic mutation will not saturate when the discriminator rejects the generated samples. Thus, the heuristic mutation avoids vanishing gradient and provides useful generator updates (Fig. 2). However, according to [24], given the optimal discriminator D^*, minimizing the heuristic mutation is equal to minimizing $[\text{KL}(p_g||p_{\text{data}}) - 2\text{JSD}(p_g||p_{\text{data}})]$, i.e., inverted KL minus two JSDs. Intuitively, the JSD sign is negative, which means pushing these two distributions away from each other. In practice, this may lead to training instability and generative quality fluctuations [44].

3) Least-Squares Mutation: The least-squares mutation is inspired by LSGAN [20], where the least-squares objectives are utilized to penalize its generator to deceive the discriminator. In this paper, we formulate the least-squares mutation as

$$\mathcal{M}_G^{\text{least-square}} = \mathbb{E}_{z\sim p_z}\Big[(D(G(z)) - 1)^2\Big]. \tag{5}$$

As shown in Fig. 2, the least-squares mutation is nonsaturating when the discriminator can recognize the generated sample [i.e., $D(G(z)) \to 0$]. When the discriminator output grows, the least-squares mutation saturates, eventually approaching zero. Therefore, similar to the heuristic mutation, the least-squares mutation can avoid vanishing gradient when the discriminator has a significant advantage over the generator. Meanwhile, compared to the heuristic mutation, although the least-squares mutation will not assign an extremely high cost to generate fake samples, it will also not assign an extremely low cost to mode dropping,[2] which partly avoids mode collapse [20].

[2]Arjovsky and Bottou [24] demonstrated that the heuristic objective suffers from mode collapse since $\text{KL}(p_g||p_{\text{data}})$ assigns a high cost to generating fake samples but an extremely low cost to mode dropping.

Note that, different from GAN-minimax and GAN-heuristic, LSGAN employs a different objective ("least squares") to optimize the discriminator, i.e.,

$$\begin{aligned}\mathcal{L}_D^{\text{LSGAN}} &= \frac{1}{2}\mathbb{E}_{x\sim p_{\text{data}}}\Big[(D(x)-1)^2\Big] + \frac{1}{2}\mathbb{E}_{z\sim p_z}\Big[D(G(z))^2\Big] \\ &= \int_x \frac{1}{2}\Big(p_{\text{data}}(x)(D(x)-1)^2 + p_g(x)D(x)^2\Big)dx. \end{aligned}\tag{6}$$

Yet, with respect to $D(x)$, the function $\mathcal{L}_D^{\text{LSGAN}}$ achieves its minimum in $[0, 1]$ at $([p_{\text{data}}(x)]/[p_{\text{data}}(x) + p_g(x)])$, which is equivalent to ours [i.e., (2)].

Therefore, although we employ only one discriminator as the environment to distinguish real and generated samples, it is sufficient to provide adaptive losses for all mutations described above.

D. Evaluation

In an evolutionary algorithm, evaluation is the operation of measuring the quality of individuals. To determine the evolutionary direction (i.e., individuals' selection), we devise an evaluation (or fitness) function to measure the performance of evolved individuals (i.e., children). Typically, we mainly focus on two properties: 1) the quality and 2) the diversity of generated samples. Quality is measured for each generated sample. If a generated image could be realistic enough, it will fool the discriminator. On the other hand, the diversity measures whether the generator could spread the generated samples out enough, which could largely avoid mode collapse.

First, we simply feed generator produced images into the discriminator D and observe the average value of the output, which we name the *quality fitness score*

$$\mathcal{F}_q = \mathbb{E}_z[D(G(z))]. \tag{7}$$

Note that discriminator D is constantly upgraded to be optimal during the training process, reflecting the quality of generators at each evolutionary (or adversarial) step. If a generator obtains a relatively high-quality score, its generated samples can deceive the discriminator and the generated distribution is further approximate to the data distribution.

Besides generative quality, we also pay attention to the diversity of generated samples and attempt to overcome the mode collapse issue in GAN optimization. Recently, Nagarajan and Kolter [25] proposed a gradient-based regularization term to stabilize the GAN optimization and suppress mode collapse. When the generator collapses to a small region, the discriminator will subsequently label collapsed points as fake with obvious countermeasure (i.e., big gradients). In contrast, if the generator is capable of spreading generated data out enough, the discriminator will not be much confident to label generated samples as fake data (i.e., updated with small gradients). Other techniques, e.g., exploiting conditioning latent vector [64] or subspace [65], can also be applied to purse diversity. But this issue does not fall within the scope of this paper.

We employ a similar principle to evaluate generator optimization stability and generative diversity. Here, since the gradient-norm of the discriminator could vary largely during training, we employed an logarithm to shrink its fluctuation.

Specifically, the minus log-gradient-norm of optimizing D is utilized to measure the diversity of generated samples. If an evolved generator obtains a relatively high value, which corresponds to small discriminator gradients, its generated samples tend to spread out enough, to avoid the discriminator from having obvious countermeasures. Thus, the mode collapse issue can be suppressed and the discriminator will change smoothly, which helps to improve the training stability. Formally, the *diversity fitness score* is defined as

$$\mathcal{F}_{\mathrm{d}} = -\log\left|\left|\nabla_D - \mathbb{E}_x\big[\log D(x)\big] - \mathbb{E}_z\big[\log(1 - D(G(z)))\big]\right|\right|. \tag{8}$$

Based on the aforementioned two fitness scores, we can finally give the evaluation (or fitness) function of the proposed evolutionary algorithm

$$\mathcal{F} = \mathcal{F}_{\mathrm{q}} + \gamma \mathcal{F}_d \tag{9}$$

where $\gamma \geq 0$ balances two measurements: 1) generative quality and 2) diversity. Overall, a relatively high fitness score $\mathcal{F}$, leads to higher training efficiency and better generative performance.

It is interesting to note that, the proposed fitness score can also be regarded as an objective function for generator G. However, as demonstrated in our experiments, without the proposed evolutionary strategy, this fitness objective is still hard to achieve a convincing performance. Meanwhile, similar with most existing objective functions of GANs, the fitness function will continuously fluctuate during the dynamic adversarial training process. Specifically, in each iteration, the fitness score of generators are evaluated based on the current discriminator. Comparing them, we are capable of selecting well-performing ones. However, since the discriminator will be updated according to different generators, it is hard to discuss the connection between the fitness scores over different iterations.

E. Selection

In an evolutionary algorithm, the counterpart of the mutation operators is the selection. In the proposed E-GAN, we employ a simple yet useful survivor selection strategy to determine the next generation based on the fitness score of existing individuals.

First of all, we should notice that both the generators (i.e., population) and the discriminator (i.e., environment) are optimized alternately in a dynamic procedure. Thus, the fitness function is not fixed and the fitness score of generators can only be evaluated by the corresponding discriminator in the same evolutionary generation, which means fitness scores evaluated in different generations cannot compare with each other. In addition, due to the mutation operators of the proposed E-GAN actually correspond to different adversarial training objectives, selecting desired offspring is equivalent to selecting the effective adversarial strategies. During the adversarial process, we hope that generator(s) can do it best to fool the discriminator (i.e., implement the optimal adversarial strategy). Considering these two points, we utilize the comma selection, i.e., (μ, λ)-selection [66] as the selection mechanism of E-GAN. Specifically, after sorting the current offspring population $\{\mathbf{x}_i\}_{i=1}^{\lambda}$ according to their fitness scores $\mathcal{F}_i$, the μ-best individuals are selected to form the next generation.

Algorithm 1 E-GANs. Default Values $\alpha = 0.0002$, $\beta_1 = 0.5$, $\beta_2 = 0.99$, $n_D = 3$, $n_m = 3$, and $m = 32$

Require: the batch size m. the discriminator's updating steps per iteration n_D. the number of parents μ. the number of mutations n_m. Adam hyper-parameters α, β_1, β_2, the hyper-parameter γ of evaluation function.
Require: initial discriminator's parameters w_0. initial generators' parameters $\{\theta_0^1, \theta_0^2, \ldots, \theta_0^{\mu}\}$.
for number of training iterations **do**
  for $k = 0, \ldots, n_D$ **do**
    Sample a batch of $\{x^{(i)}\}_{i=1}^{m} \sim p_{\mathrm{data}}$ (training data), and a batch of $\{z^{(i)}\}_{i=1}^{m} \sim p_z$ (noise samples).
    $g_w \leftarrow \nabla_w[\frac{1}{m}\sum_{i=1}^{m} \log D_w(x^{(i)})$
      $+\frac{1}{m}\sum_{j=1}^{\mu}\sum_{i=1}^{m/\mu} \log(1 - D_w(G_{\theta^j}(z^{(i)})))]$
    $w \leftarrow \mathrm{Adam}(g_w, w, \alpha, \beta_1, \beta_2)$
  end for
  for $j = 0, \ldots, \mu$ **do**
    for $h = 0, \ldots, n_m$ **do**
      Sample a batch of $\{z^{(i)}\}_{i=1}^{m} \sim p_z$
      $g_{\theta^{j,h}} \leftarrow \nabla_{\theta^j} \mathcal{M}_G^h(\{z^{(i)}\}_{i=1}^{m}, \theta^j)$
      $\theta_{\mathrm{child}}^{j,h} \leftarrow \mathrm{Adam}(g_{\theta^{j,h}}, \theta^j, \alpha, \beta_1, \beta_2)$
      $\mathcal{F}^{j,h} \leftarrow \mathcal{F}_q^{j,h} + \gamma \mathcal{F}_d^{j,h}$
    end for
  end for
  $\{\mathcal{F}^{j_1,h_1}, \mathcal{F}^{j_2,h_2}, \ldots\} \leftarrow \mathrm{sort}(\{\mathcal{F}^{j,h}\})$
  $\theta^1, \theta^2, \ldots, \theta^{\mu} \leftarrow \theta_{\mathrm{child}}^{j_1,h_1}, \theta_{\mathrm{child}}^{j_2,h_2}, \ldots, \theta_{\mathrm{child}}^{j_{\mu},h_{\mu}}$
end for

F. E-GAN

Having introduced the proposed evolutionary algorithm and corresponding mutation operations, evaluation criteria and selection strategy, the complete E-GAN training process is concluded in Algorithm 1. Overall, in E-GAN, generators $\{G\}$ are regarded as an evolutionary population and discriminator D acts as an environment. For each evolutionary step, generators are updated with different mutations (or objectives) to accommodate the current environment. According to the principle of survival of the fittest, only well-performing children will survive and participate in future adversarial training. Unlike the two-player game with a fixed and static adversarial training objective in conventional GANs, E-GAN allows the algorithm to integrate the merits of different adversarial objectives and generate the most competitive solution. Thus, during training, the evolutionary algorithm not only largely suppresses the limitations (vanishing gradient, mode collapse, etc.) of individual adversarial objectives, but it also harnesses their advantages to search for a better solution.

IV. Experiments

To evaluate the proposed E-GAN, we run experiments on serval generative tasks and present the experimental results

TABLE I
ARCHITECTURES OF THE GENERATIVE AND DISCRIMINATIVE NETWORKS

Generative network G
Input: Noise z, 100
[layer 1] Fully connect and Reshape to $(4 \times 4 \times 2048)$; *LReLU*;
[layer 2] Transposed Conv. (4, 4, 2048), stride=2; *LReLU*;
[layer 3] Transposed Conv. (4, 4, 1024), stride=1; *LReLU*;
[layer 4] Transposed Conv. (4, 4, 1024), stride=2; *LReLU*;
[layer 5] Transposed Conv. (4, 4, 512), stride=1; *LReLU*;
[layer 6] Transposed Conv. (4, 4, 512), stride=2; *LReLU*;
[layer 7] Transposed Conv. (4, 4, 256), stride=1; *LReLU*;
[layer 8] Transposed Conv. (4, 4, 256), stride=2; *LReLU*;
[layer 9] Transposed Conv. (4, 4, 128), stride=1; *LReLU*;
[layer 10] Transposed Conv. (4, 4, 128), stride=2; *LReLU*;
[layer 11] Transposed Conv. (3, 3, 3), stride=1; *Tanh*;
Output: Generated Image, $(128 \times 128 \times 3)$
Discriminative network D
Input: Image $(128 \times 128 \times 3)$
[layer 1] Conv. (3, 3, 128), stride=1; Batchnorm; *LReLU*;
[layer 2] Conv. (3, 3, 128), stride=2; Batchnorm; *LReLU*;
[layer 3] Conv. (3, 3, 256), stride=1; Batchnorm; *LReLU*;
[layer 4] Conv. (3, 3, 256), stride=2; Batchnorm; *LReLU*;
[layer 5] Conv. (3, 3, 512), stride=1; Batchnorm; *LReLU*;
[layer 6] Conv. (3, 3, 512), stride=2; Batchnorm; *LReLU*;
[layer 7] Conv. (3, 3, 1024), stride=1; Batchnorm; *LReLU*;
[layer 8] Conv. (3, 3, 1024), stride=2; Batchnorm; *LReLU*;
[layer 9] Conv. (3, 3, 2048), stride=1; Batchnorm; *LReLU*;
[layer 10] Conv. (3, 3, 2048), stride=2; Batchnorm; *LReLU*;
[layer 11] Fully connected (1); *Sigmoid*;
Output: Real or Fake (Probability)

in this section. Compared with some previous GAN methods, we show that the proposed E-GAN can achieve impressive generative performance on large-scale image datasets.

A. Implementation Details

We evaluate E-GAN on two synthetic datasets and three image datasets: 1) CIFAR-10 [67]; 2) LSUN bedroom [68]; and 3) CelebA [69]. For fair comparisons, we adopted the same network architectures with existing works [22], [44]. In addition, to achieve better performance on generating 128×128 images, we slightly modified both the generative network and the discriminator network based on the DCGAN architecture. Specifically, the batch norm layers are removed from the generator, and more features channels are applied to each convolutional layers. The detailed networks are listed in Table I, note that the network architectures of the other comparison experiments can be easily found in the referenced works.

We use the default hyper-parameter values listed in Algorithm 1 for all experiments. Note that the hyper-parameter γ is utilized to balance the measurements of samples quality (i.e., $\mathcal{F}_q$) and diversity (i.e., $\mathcal{F}_d$). Usually, the quality fitness score $\mathcal{F}_q$ lies in $[0, 1]$, while the diversity fitness score $\mathcal{F}_d$ measures the log-gradient-norm of the discriminator D, which can vary largely according to D's scale. Therefore, we first determine γ's range based on the selected discriminator D. Then, we run grid search to find its value. In practice, we choose $\gamma = 0.5$ for the synthetic datasets, and $\gamma = 0.001$ for real-world data. In addition, recently, some works [44], [70] proposed the gradient penalty (GP) term to regularize the discriminator to provide precise gradients for updating the generator. Within the adversarial training framework, our contributions are largely orthogonal to the GP term. In our experiments, through the setting without GP term, we demonstrated the efficiency of the proposed method. Then, after introducing the GP term, the generative performance was further improved, which demonstrated our framework could also benefit from the regularization technique for the discriminator. Furthermore, all experiments were trained on Nvidia Tesla V100 GPUs. To train a model for 64×64 images using the DCGAN architecture cost around 20 h on a single GPU.

B. Evaluation Metrics

Besides directly reported generated samples of the learned generative networks, we choose the maximum mean discrepancy (MMD) [71], [72], the inception score (IS) [43], and the Fréchet Inception distance (FID) [73] as quantitative metrics. Among them, the MMD can be utilized to measure the discrepancy between the generated distribution and the target distribution for synthetic Gaussian mixture datasets. However, the MMD is difficult to directly apply to high-dimensional image datasets. Therefore, through applying the pretrained Inception v3 network [74] to generated images, the IS computes the KL divergence between the conditional class distribution and the marginal class distribution. Usually, this score correlates well with the human scoring of the realism of generated images from the CIFAR-10 dataset, and a higher value indicates better image quality. However, some recent works [6], [75] revealed serious limitations of the IS, e.g., the target data distribution (i.e., the training data) has not been considered. In our experiments, we utilize the IS to measure the E-GAN's performance on the CIFAR-10, and compare our results to previous works. Moreover, FID is a more principled and reliable metric and has demonstrated better correlations with human evaluation for other datasets. Specifically, FID calculates the Wasserstein-2 distance between the generated images and the real-world images in the high-level feature space of the pretrained Inception v3 network. Note that lower FID means closer distances between the generated distribution and the real-world data distribution. In all experiments, we randomly generated 50k samples to calculate the MMD, IS, and FID.

C. Synthetic Datasets and Mode Collapse

In the first experiment, we adopt the experimental design proposed in [76], which trains GANs on 2-D Gaussian mixture distributions. The mode collapse issue can be accurately measured on these synthetic datasets, since we can clearly

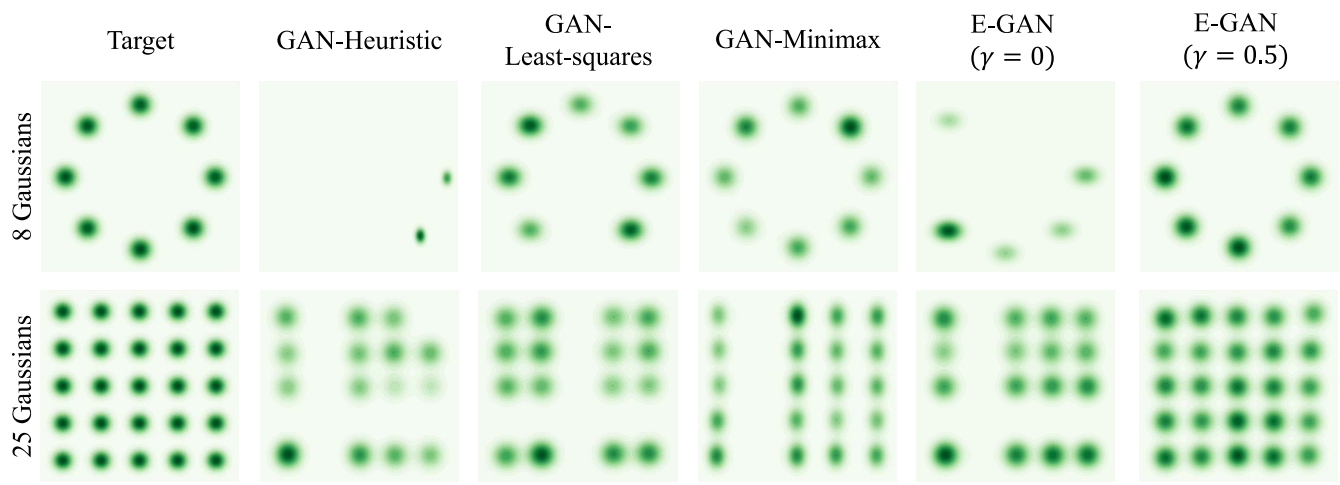

Fig. 3. KDE plots of the target data and generated data from different GANs trained on mixtures of Gaussians. In the first row, a mixture of 8 Gaussians arranged in a circle. In the second row, a mixture of 25 Gaussians arranged in a grid.

TABLE II
MMD ($\times 10^{-2}$) WITH MIXED GAUSSIAN DISTRIBUTIONS ON OUR TOY DATASETS. WE RAN EACH METHOD FOR TEN TIMES, AND REPORT THEIR AVERAGE AND BEST RESULTS. THE METHOD WITH LOWER MMD VALUE IMPLIES THE GENERATED DISTRIBUTION IS CLOSER TO THE TARGET ONE

Methods	8 Gaussians		25 Gaussians	
	Average	Best	Average	Best
GAN-Heuristic	45.27	33.2	2.80	2.19
GAN-Least-squares	3.99	3.16	1.83	1.72
GAN-Minimax	2.94	1.89	1.65	1.55
E-GAN ($\lambda = 0$, without GP)	11.54	7.31	1.69	1.60
E-GAN ($\lambda = 0.5$, without GP)	**2.36**	**1.17**	**1.20**	**1.04**

TABLE III
INCEPTION SCORES AND FIDS WITH UNSUPERVISED IMAGE GENERATION ON CIFAR-10. THE METHOD WITH HIGHER IS OR LOWER FID IMPLIES THE GENERATED DISTRIBUTION IS CLOSER TO THE TARGET ONE. † [22], ‡ [6]

Methods	Inception score	FID
Real data	$11.24 \pm .12$	7.8
-Standard CNN-		
(ours) E-GAN-GP ($\mu = 1$)	$7.13 \pm .07$	33.2
(ours) E-GAN-GP ($\mu = 2$)	$7.23 \pm .08$	31.6
(ours) E-GAN-GP ($\mu = 4$)	$7.32 \pm .09$	29.8
(ours) E-GAN-GP ($\mu = 8$)	$\mathbf{7.34 \pm .07}$	**27.3**
(ours) E-GAN ($\mu = 1$, without GP)	$6.98 \pm .09$	36.2
DCGAN (without GP)†	$6.64 \pm .14$	-
GAN-GP‡	$6.93 \pm .08$	37.7
WGAN-GP‡	$6.68 \pm .06$	40.2

observe and measure the generated distribution and the target data distribution. As shown in Fig. 3, we employ two challenging distributions to evaluate E-GAN, a mixture of 8 Gaussians arranged in a circle and a mixture of 25 Gaussians arranged in a grid.[3] Here, to evaluate if the proposed diversity fitness score can reduce the mode collapse, we did not introduce the GP norm and set the survived parents number μ as 1, i.e., during each evolutionary step, only the best candidature are kept.

First, we utilize existing individual adversarial objectives (i.e., conventional GANs) to perform the adversarial training process. We train each method 50K iterations and report the Kernel density estimation (KDE) plots in Fig. 3. Meanwhile, the average and the best MMD values which running each method ten times are reported in Table II. The results show that all of the individual adversarial objectives suffer from mode collapse to a greater or lesser degree. Then, we set hyper-parameter γ as 0 and test the proposed E-GAN, which means the diversity fitness score was not considered during the training process. The results show that the evolutionary framework still has problems with the mode collapse. However, when the diversity fitness score is considered in the selection stage (in this experiment, we set $\gamma = 0.5$), the mode collapse issue is largely suppressed and the trained generator can more accurately fit the target distributions. This demonstrates that our diversity fitness score has the capability of measuring the sample diversity of updated generators and further suppress the mode collapse problem.

[3]We obtain both 2-D distributions and network architectures from the code provided in [44].

D. CIFAR-10 and Training Stability

In the proposed E-GAN, we utilize the evolutionary algorithm with different mutations (i.e., different updating strategies) to optimize generator(s) $\{G\}$. To demonstrate the advantages of the proposed evolutionary algorithm over existing two-player adversarial training strategies (i.e., updating generator with a single objective), we train these methods on CIFAR-10 and plot inception scores [43] over the training process. For a fair comparison, we did not introduce the GP norm into our E-GAN and set the parents number μ

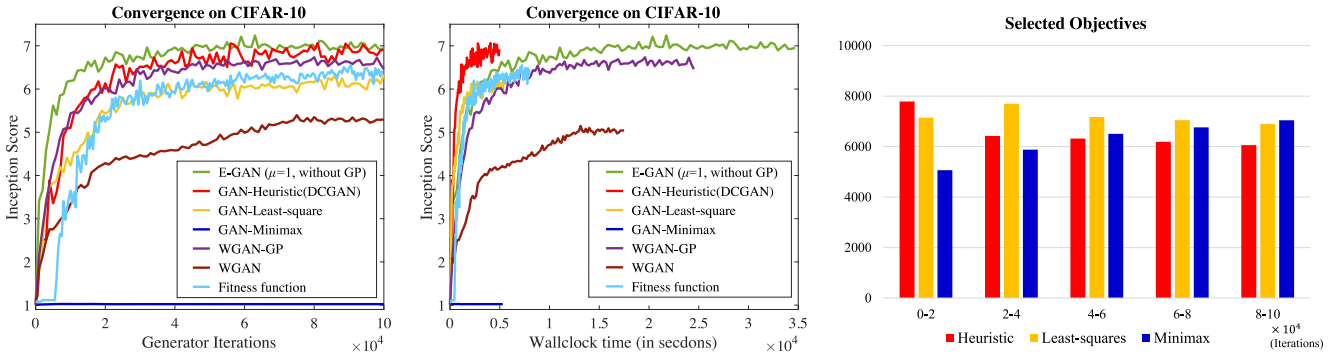

Fig. 4. Experiments on the CIFAR-10 dataset. CIFAR-10 IS over generator iterations (left), over wall-clock time (middle), and the graph of selected mutations in the E-GAN training process (right).

TABLE IV
FIDs WITH UNSUPERVISED IMAGE GENERATION ON LSUN BEDROOM DATASET. THE METHOD WITH LOWER FID IMPLIES THE GENERATED DISTRIBUTION IS CLOSER TO THE TARGET ONE

Methods	FID			
	-Baseline-	**-Weak G-**	**-Weak D-**	**-Weak both-**
DCGAN	43.7	187.3	410.6	82.7
LSGAN	46.3	452.9	423.1	126.2
WGAN	51.1	113.6	129.2	115.7
WGAN-GP	38.5	66.7	385.8	73.2
(ours) E-GAN ($\mu = 1$, without GP)	34.2	63.3	64.8	71.9
(ours) E-GAN ($\mu = 4$, without GP)	**29.7**	**59.1**	**55.2**	**60.9**

as 1. Moreover, the same network architecture is used for all methods.

As shown in Fig. 4(left), E-GAN can get higher IS with less training steps. Meanwhile, E-GAN also shows comparable stability when it goes to convergence. By comparison, conventional GAN objectives expose their different limitations, such as instability at convergence (GAN-Heuristic), slow convergence (GAN-Least square), and invalid (GAN-minimax). In addition, we employ the proposed fitness function [i.e., (9)] as generator's objective function, and find its performance is also inferior to E-GAN. This experiment further demonstrates the advantages of the proposed evolutionary framework. Through creating and selecting from multiple candidates, the evolutionary framework can leverage strengths of different objective functions (i.e., different distances) to accelerate the training process and improve the generative performance. Based on the evolutionary framework, E-GAN not only overcome the inherent limitations of these individual adversarial objectives, but it also outperforms other GANs (the WGAN and its improved variant WGAN-GP). Furthermore, when we only keep one parent during each evolutionary step, E-GAN achieves comparable convergence speed in terms of wall-clock time [Fig. 4(middle)]. During training E-GAN, we recorded the selected objective in each step [Fig. 4(right)]. At the beginning of training, the heuristic objective and the least-square objective are selected more frequently than the minimax objective. It may be due to the fact that the minimax objective is hard to provide effective gradients (i.e., vanishing gradient) when the discriminator can easily recognize generated samples. Along with the generator approaching convergence (after 20K steps), ever more minimax objectives are employed, yet the number of selected heuristic objectives is falling. As aforementioned, the minus JSDs of the heuristic objective may tend to push the generated distribution away from target data distribution and lead to training instability. However, in E-GAN, beyond the heuristic objective, we have other options of objective, which improves the stability at convergence.

Furthermore, we discussed the relationship between the survived parents' number and generative performance. As shown in Table III, both the IS and FID are utilized to evaluate the generative performance of learned generators. First, compared with the basic E-GAN (i.e., E-GAN, $\mu = 1$, without GP), adding the GP norm to optimize the discriminator indeed improves generative performance. Then, we preserved multiple parents during the E-GAN training process and measured their scores at the end of training. We can easily observe that the generative performance becomes better accompanied with keeping more parents during the training. This further demonstrates the proposed evolutionary learning paradigm could suppress the unstable and large-scale optimization problems of GANs.

Theoretically, if we regard updating and evaluating a child G as an operation and define mutations number as n, keeping p parents will cost $O(np)$ operations in each iteration. Comparing with traditional GANs, our evolutionary framework would cost more time in each iteration. However, since the evolutionary strategy always preserves

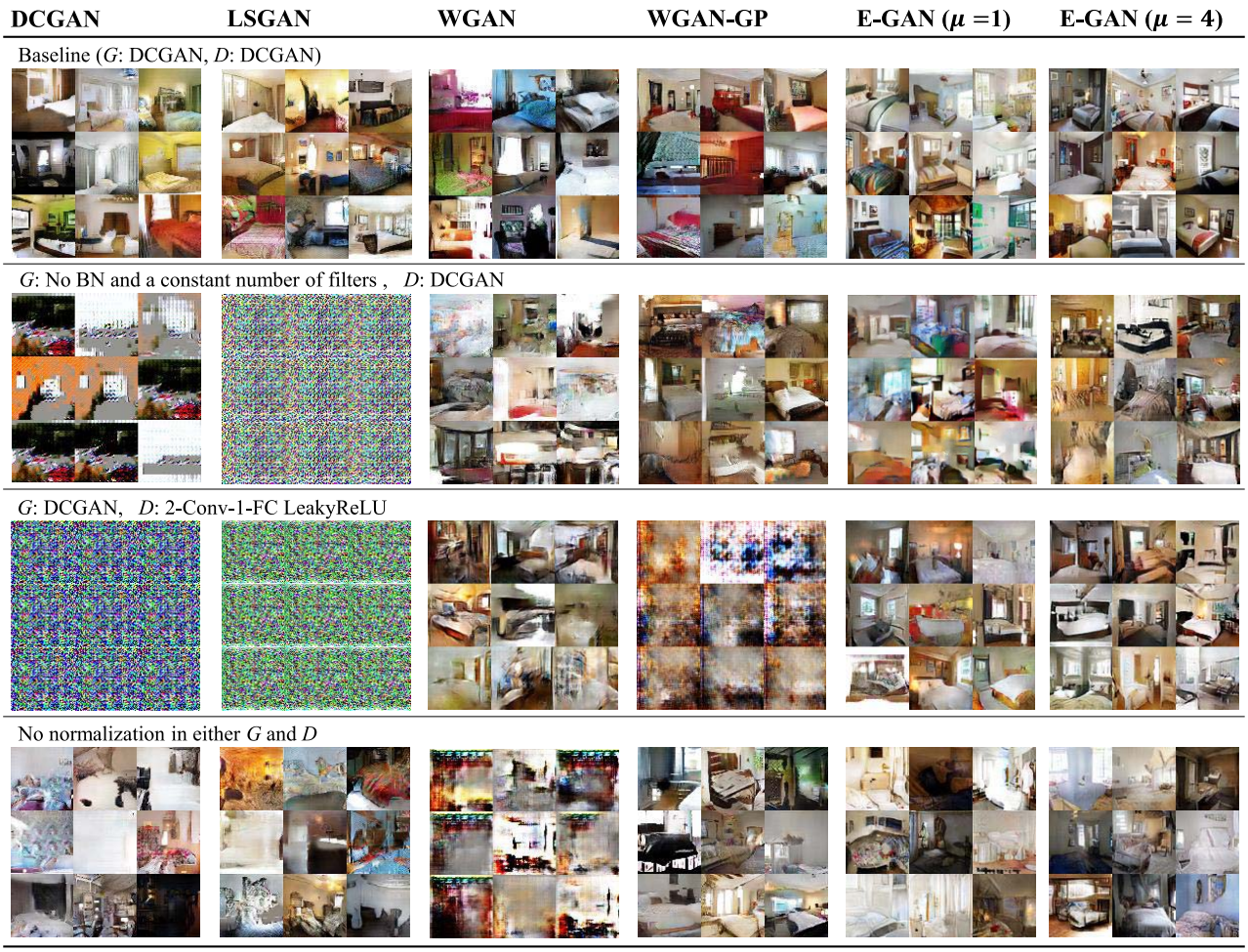

Fig. 5. Experiments of architecture robustness. Different GAN architectures, which correspond to different training challenges, trained with six different GAN methods (or settings). The proposed E-GAN achieved promising performance under all architecture settings.

Fig. 6. Generated bedroom images on 128 × 128 LSUN bedrooms.

Fig. 7. Generated human face images on 128 × 128 CelebA dataset.

the well-performing off-spring, to achieve the same generative performance, E-GAN usually spends less training steps [Fig. 4(left)]. Overall, keeping one parent during each evolutionary step will only slightly reduce the time-efficiency but with better performance [Fig. 4(middle)]. Yet, accompanied by increasing p, although the generative performance can be further improved, it will also cost more time on training the E-GAN model. Here, if we regard the parents number p as a hyper-parameter of our algorithm, we found setting its value less than or equal to 4 is a preferable choice.

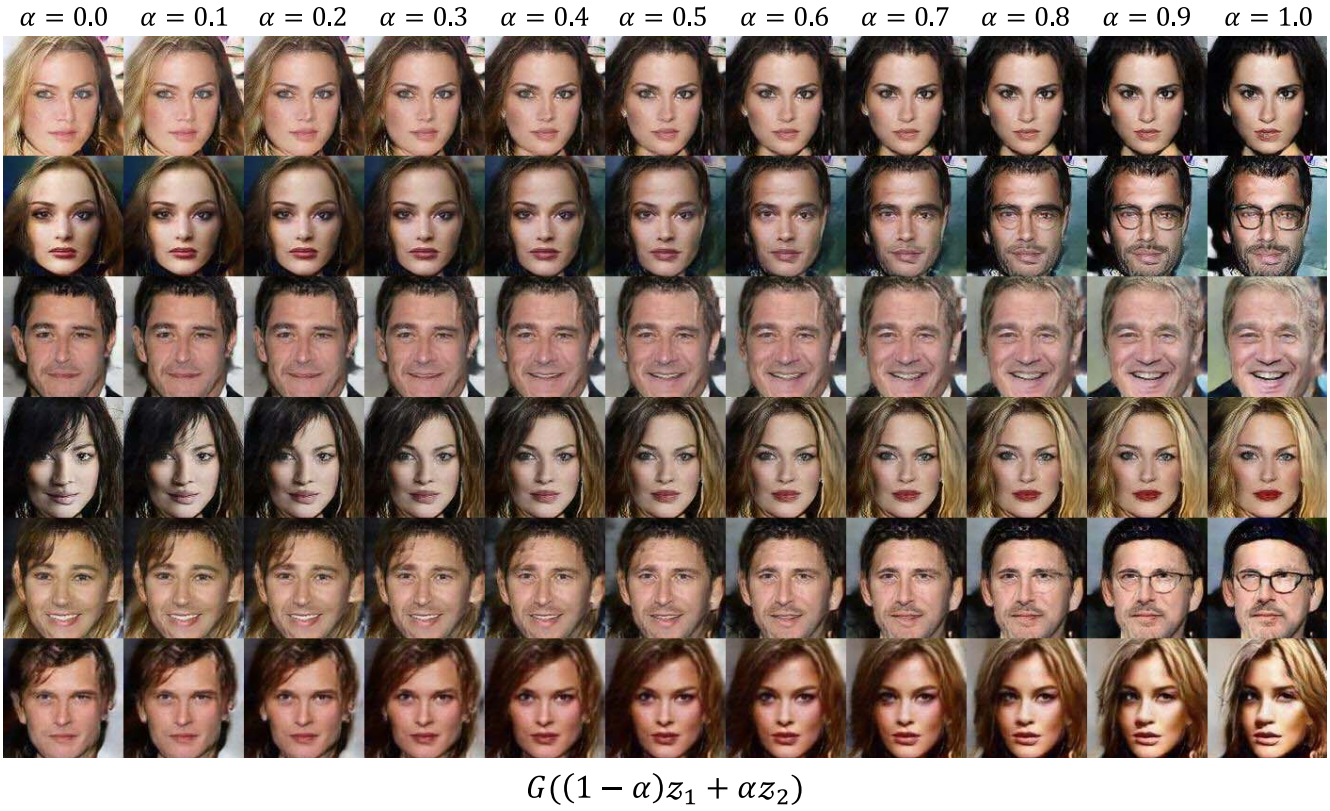

Fig. 8. Interpolating in latent space. For selected pairs of generated images from a well-trained E-GAN model, we record their latent vectors z_1 and z_2. Then, samples between them are generated by linear interpolation between these two vectors.

Within this interval, we can easily improve the generative performance by sacrificing affordable computation cost. If we continually increase the number of survivors, the generative performance can only be improved mildly yet largely reduce the training efficiency. In practice, we need to further balance the algorithm efficiency and performance according to different situations.

E. LSUN and Architecture Robustness

The architecture robustness is another advantage of E-GAN. To demonstrate the training stability of our method, we train different network architectures on the LSUN bedroom dataset [68] and compare with several existing works. In addition to the baseline DCGAN architecture, we choose three additional architectures corresponding to different training challenges: 1) limiting the recognition capability of the discriminator *D*, i.e., 2-Conv-1-FC LeakyReLU discriminator (abbreviated as weak *D*); 2) limiting the expression capability of the generator *G*, i.e., no batchnorm and a constant number of filters in the generator (weak *G*); and 3) reducing the network capability of the generator and discriminator together, i.e., remove the BN in both the generator *G* and discriminator *D* (weak both). For each architecture, we test six different methods (or settings): 1) DCGAN; 2) LSGAN; 3) original WGAN (with weight clipping); 4) WGAN-GP; 5) our E-GAN ($\mu = 1$); and 6) E-GAN ($\mu = 4$). For each method, we used the default configurations recommended in the respective studies (these methods are summarized in [44]) and train each model for 100K iterations. Some generated samples are reported in Fig. 5, and the quantitative results (i.e., FID) are listed in Table IV. Through observation, we find that all of these GAN methods achieved promising performance with the baseline architecture. For DCGAN and LSGAN, when the balance between the generator and discriminator is broken (i.e., only one of them is limited), these two methods have difficulty generating any reasonable samples. Meanwhile, we find the performance of the standard WGAN (with weight clipping) is mostly decided by the generator *G*. When we limit *G*'s capability, the generative performance is largely reduced. Regarding the WGAN-GP, we find that the generative performance may mainly depend on the discriminator (or critic). Our E-GAN achieved promising results under all architecture settings. Moreover, we again demonstrated that the model performance is growing with the number of survived parents.

Furthermore, we trained E-GAN to generate higher resolution (128×128) bedroom images (Fig. 6). Observing generated images, we demonstrate that E-GAN can be trained to generate diversity and high-quality images from the target data distribution.

F. CelebA and Space Continuity

Besides the LSUN bedroom dataset, we also train our E-GAN using the aligned faces from CelebA dataset. Since humans excel at identifying facial flaws, generating high-quality human face images is challenging. Similar to

generating bedrooms, we employ the same architectures to generate 128×128 RGB human face images (Fig. 7). In addition, given a well-trained generator, we evaluate the performance of the embedding in the latent space of noisy vectors z. In Fig. 8, we first select pairs of generated faces and record their corresponding latent vectors z_1 and z_2. The two images in one pair have different attributes, such as gender, expression, hairstyle, and age. Then, we generate novel samples by linear interpolating between these pairs (i.e., corresponding noisy vectors). We find that these generated samples can seamlessly change between these semantically meaningful face attributes. This experiment demonstrates that generator training does not merely memorize training samples but learns a meaningful projection from latent noisy space to face images. Meanwhile, it also shows that the generator trained by E-GAN does not suffer from mode collapse, and shows great space continuity. Overall, during the GAN training process, the training stability is easily influenced by "bad" updating, which could lead the generated samples to low quality or lacking diversity, while the proposed evolutionary mechanism largely avoids undesired updating and promotes the training to an ideal direction.

V. Conclusion

In this paper, we presented an E-GAN framework for training deep generative models. To reduce training difficulties and improve generative performance, we devised an evolutionary algorithm to evolve a population of generators to adapt to the dynamic environment (i.e., the discriminator D). In contrast to conventional GANs, the evolutionary paradigm allows the proposed E-GAN to overcome the limitations of individual adversarial objectives and preserve the well-performing offspring after each iteration. Experiments showed that E-GAN improves the training stability of GAN models and achieves convincing performance in several image generation tasks. In this paper, we mainly contribute to improving the image generation performance. More generation tasks will be considered in future works, such as video generation [33] and text generation [77].

References

[1] I. Goodfellow *et al.*, "Generative adversarial nets," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, 2014, pp. 2672–2680.

[2] X. Chen *et al.*, "InfoGAN: Interpretable representation learning by information maximizing generative adversarial nets," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, 2016, pp. 2172–2180.

[3] Z. Gan *et al.*, "Triangle generative adversarial networks," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, 2017, pp. 5253–5262.

[4] H. Zhang *et al.*, "StackGAN: Text to photo-realistic image synthesis with stacked generative adversarial networks," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, 2017, pp. 5908–5916.

[5] T. Karras, T. Aila, S. Laine, and J. Lehtinen, "Progressive growing of GANs for improved quality, stability, and variation," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2018, pp. 1–26.

[6] T. Miyato, T. Kataoka, M. Koyama, and Y. Yoshida, "Spectral normalization for generative adversarial networks," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2018, pp. 1–26.

[7] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2017, pp. 5967–5976.

[8] C. Wang, C. Wang, C. Xu, and D. Tao, "Tag disentangled generative adversarial networks for object image re-rendering," in *Proc. 26th Int. Joint Conf. Artif. Intell. (IJCAI)*, 2017, pp. 2901–2907.

[9] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, 2017, pp. 2242–2251.

[10] T.-C. Wang *et al.*, "High-resolution image synthesis and semantic manipulation with conditional GANs," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2018, pp. 8798–8807.

[11] C. Vondrick, H. Pirsiavash, and A. Torralba, "Generating videos with scene dynamics," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, 2016, pp. 613–621.

[12] C. Finn, I. Goodfellow, and S. Levine, "Unsupervised learning for physical interaction through video prediction," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, 2016, pp. 64–72.

[13] C. Vondrick and A. Torralba, "Generating the future with adversarial transformers," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Honolulu, HI, USA, 2017, pp. 2992–3000.

[14] Y. Zhang, Z. Gan, and L. Carin, "Generating text via adversarial training," in *Proc. NIPS Workshop Adversarial Training*, 2016, pp. 1–6.

[15] W. Fedus, I. Goodfellow, and A. M. Dai, "MaskGAN: Better text generation via filling in the___," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2018, pp. 1–17.

[16] J. Lu, A. Kannan, J. Yang, D. Parikh, and D. Batra, "Best of both worlds: Transferring knowledge from discriminative learning to a generative visual dialog model," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, 2017, pp. 313–323.

[17] X. Lan, A. J. Ma, and P. C. Yuen, "Multi-cue visual tracking using robust feature-level fusion based on joint sparse representation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2014, pp. 1194–1201.

[18] X. Chen, C. Xu, X. Yang, L. Song, and D. Tao, "Gated-GAN: Adversarial gated networks for multi-collection style transfer," *IEEE Trans. Image Process.*, vol. 28, no. 2, pp. 546–560, Feb. 2019.

[19] S. Arora, R. Ge, Y. Liang, T. Ma, and Y. Zhang, "Generalization and equilibrium in generative adversarial nets (GANs)," in *Proc. 34th Int. Conf. Mach. Learn. (ICML)*, 2017, pp. 224–232.

[20] X. Mao *et al.*, "Least squares generative adversarial networks," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, 2017, pp. 2813–2821.

[21] J. Zhao, M. Mathieu, and Y. LeCun, "Energy-based generative adversarial network," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2017, pp. 1–17.

[22] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2016, pp. 1–16.

[23] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein generative adversarial networks," in *Proc. 34th Int. Conf. Mach. Learn. (ICML)*, 2017, pp. 214–223.

[24] M. Arjovsky and L. Bottou, "Towards principled methods for training generative adversarial networks," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2017, pp. 1–17.

[25] V. Nagarajan and J. Z. Kolter, "Gradient descent GAN optimization is locally stable," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, 2017, pp. 5591–5600.

[26] C. Qian, J.-C. Shi, K. Tang, and Z.-H. Zhou, "Constrained monotone k-submodular function maximization using multiobjective evolutionary algorithms with theoretical guarantee," *IEEE Trans. Evol. Comput.*, vol. 22, no. 4, pp. 595–608, Aug. 2018.

[27] S. He *et al.*, "Cooperative co-evolutionary module identification with application to cancer disease module discovery," *IEEE Trans. Evol. Comput.*, vol. 20, no. 6, pp. 874–891, Dec. 2016.

[28] Y. Du, M.-H. Hsieh, T. Liu, and D. Tao. (2018). *The Expressive Power of Parameterized Quantum Circuits*. [Online]. Available: https://arxiv.org/abs/1810.11922

[29] L. M. Antonio and C. A. Coello Coello, "Coevolutionary multiobjective evolutionary algorithms: A survey of the state-of-the-art," *IEEE Trans. Evol. Comput.*, vol. 22, no. 6, pp. 851–865, Dec. 2018.

[30] E. Real *et al.*, "Large-scale evolution of image classifiers," in *Proc. 34th Int. Conf. Mach. Learn. (ICML)*, 2017, pp. 214–223.

[31] T. Salimans, J. Ho, X. Chen, and I. Sutskever. (2017). *Evolution Strategies as a Scalable Alternative to Reinforcement Learning*. [Online]. Available: https://arxiv.org/abs/1703.03864

[32] I. Goodfellow. (2016). *NIPS 2016 Tutorial: Generative Adversarial Networks*. [Online]. Available: https://arxiv.org/abs/1701.00160

[33] S. Tulyakov, M.-Y. Liu, X. Yang, and J. Kautz, "MoCoGAN: Decomposing motion and content for video generation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2018, pp. 1526–1535.

[34] Y. Song *et al.*, "VITAL: Visual tracking via adversarial learning," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2018, pp. 8990–8999.

[35] X. Lan, S. Zhang, P. C. Yuen, and R. Chellappa, "Learning common and feature-specific patterns: A novel multiple-sparse-representation-based tracker," *IEEE Trans. Image Process.*, vol. 27, no. 4, pp. 2022–2037, Apr. 2018.

[36] X. Lan, A. J. Ma, P. C. Yuen, and R. Chellappa, "Joint sparse representation and robust feature-level fusion for multi-cue visual tracking," *IEEE Trans. Image Process.*, vol. 24, no. 12, pp. 5826–5841, Dec. 2015.

[37] E. Tzeng, J. Hoffman, K. Saenko, and T. Darrell, "Adversarial discriminative domain adaptation," in *Proc. Comput. Vis. Pattern Recognit. (CVPR)*, vol. 1, 2017, p. 4.

[38] Z. Qiu, Y. Pan, T. Yao, and T. Mei, "Deep semantic hashing with generative adversarial networks," in *Proc. 40th Int. ACM SIGIR Conf. Res. Dev. Inf. Retrieval*, Tokyo, Japan, 2017, pp. 225–234.

[39] E. Yang, T. Liu, C. Deng, and D. Tao, "Adversarial examples for hamming space search," *IEEE Trans. Cybern.*, to be published. doi: 10.1109/TCYB.2018.2882908.

[40] E. Yang *et al.*, "Pairwise relationship guided deep hashing for cross-modal retrieval," in *Proc. AAAI*, 2017, pp. 1618–1625.

[41] J. Donahue, P. Krähenbühl, and T. Darrell, "Adversarial feature learning," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2017, pp. 1–18.

[42] V. Dumoulin *et al.*, "Adversarially learned inference," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2017, pp. 1–18.

[43] T. Salimans *et al.*, "Improved techniques for training GANs," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, 2016, pp. 2234–2242.

[44] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. Courville, "Improved training of Wasserstein GANs," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, 2017, pp. 5769–5779.

[45] T. D. Nguyen, T. Le, H. Vu, and D. Phung, "Dual discriminator generative adversarial nets," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, 2017, pp. 2667–2677.

[46] I. Durugkar, I. Gemp, and S. Mahadevan, "Generative multi-adversarial networks," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2017, pp. 1–14.

[47] B. Neyshabur, S. Bhojanapalli, and A. Chakrabarti. (2017). *Stabilizing GAN Training With Multiple Random Projections*. [Online]. Available: https://arxiv.org/abs/1705.07831

[48] I. O. Tolstikhin, S. Gelly, O. Bousquet, C.-J. Simon-Gabriel, and B. Schölkopf, "AdaGAN: Boosting generative models," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, 2017, pp. 5424–5433.

[49] A. Ghosh, V. Kulharia, V. P. Namboodiri, P. H. S. Torr, and P. K. Dokania, "Multi-agent diverse generative adversarial networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2018, pp. 8513–8521.

[50] Q. Hoang, T. D. Nguyen, T. Le, and D. Phung, "MGAN: Training generative adversarial nets with multiple generators," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2018, pp. 1–24.

[51] K. A. De Jong, *Evolutionary Computation: A Unified Approach*. Cambridge, MA, USA: MIT Press, 2006.

[52] H.-L. Liu, L. Chen, Q. Zhang, and K. Deb, "Adaptively allocating search effort in challenging many-objective optimization problems," *IEEE Trans. Evol. Comput.*, vol. 22, no. 3, pp. 433–448, Jun. 2018.

[53] Y. Wang, M. Zhou, X. Song, M. Gu, and J. Sun, "Constructing cost-aware functional test-suites using nested differential evolution algorithm," *IEEE Trans. Evol. Comput.*, vol. 22, no. 3, pp. 334–346, Jun. 2018.

[54] P. Yang, K. Tang, and X. Yao, "Turning high-dimensional optimization into computationally expensive optimization," *IEEE Trans. Evol. Comput.*, vol. 22, no. 1, pp. 143–156, Feb. 2018.

[55] D.-C. Dang *et al.*, "Escaping local optima using crossover with emergent diversity," *IEEE Trans. Evol. Comput.*, vol. 22, no. 3, pp. 484–497, Jun. 2018.

[56] A. E. Eiben and J. Smith, "From evolutionary computation to the evolution of things," *Nature*, vol. 521, no. 7553, p. 476, May 2015.

[57] R. Miikkulainen *et al.* (2017). *Evolving Deep Neural Networks*. [Online]. Available: https://arxiv.org/abs/1703.00548

[58] S. R. Young, D. C. Rose, T. P. Karnowski, S.-H. Lim, and R. M. Patton, "Optimizing deep learning hyper-parameters through an evolutionary algorithm," in *Proc. Workshop Mach. Learn. High Perform. Comput. Environ.*, 2015, p. 4.

[59] X. Yao, "Evolving artificial neural networks," *Proc. IEEE*, vol. 87, no. 9, pp. 1423–1447, Sep. 1999.

[60] S. Lander and Y. Shang, "EvoAE—A new evolutionary method for training autoencoders for deep learning networks," in *Proc. IEEE 39th Annu. Comput. Softw. Appl. Conf. (COMPSAC)*, vol. 2, 2015, pp. 790–795.

[61] Y. Wang, C. Xu, J. Qiu, C. Xu, and D. Tao, "Towards evolutionary compression," in *Proc. 24th Int. Conf. Knowl. Disc. Data Min.*, 2018, pp. 2476–2485.

[62] R. S. Olson, N. Bartley, R. J. Urbanowicz, and J. H. Moore, "Evaluation of a tree-based pipeline optimization tool for automating data science," in *Proc. Genet. Evol. Comput. Conf.*, 2016, pp. 485–492.

[63] A. Rosales-Pérez, S. García, J. A. Gonzalez, C. A. Coello Coello, and F. Herrera, "An evolutionary multiobjective model and instance selection for support vector machines with Pareto-based ensembles," *IEEE Trans. Evol. Comput.*, vol. 21, no. 6, pp. 863–877, Dec. 2017.

[64] M. Mirza and S. Osindero. (2014). *Conditional Generative Adversarial Nets*. [Online]. Available: https://arxiv.org/abs/1411.1784

[65] J. Liang, J. Yang, H.-Y. Lee, K. Wang, and M.-H. Yang, "Sub-GAN: An unsupervised generative model via subspaces," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 698–714.

[66] O. Kramer, *Machine Learning for Evolution Strategies*, vol. 20. Cham, Switzerland: Springer, 2016.

[67] A. Krizhevsky, "Learning multiple layers of features from tiny images," Dept. Comput. Sci., Univ. Toronto, Toronto, ON, Canada, Rep. TR-2009, 2009.

[68] F. Yu *et al.* (2015). *LSUN: Construction of a Large-Scale Image Dataset Using Deep Learning With Humans in the Loop*. [Online]. Available: https://arxiv.org/abs/1506.03365

[69] Z. Liu, P. Luo, X. Wang, and X. Tang, "Deep learning face attributes in the wild," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, 2015, pp. 3730–3738.

[70] W. Fedus *et al.*, "Many paths to equilibrium: GANs do not need to decrease adivergence at every step," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2018, pp. 1–21.

[71] A. Smola, A. Gretton, L. Song, and B. Schölkopf, "A Hilbert space embedding for distributions," in *Proc. Int. Conf. Algorithmic Learn. Theory*, 2007, pp. 13–31.

[72] A. Gretton, K. M. Borgwardt, M. J. Rasch, B. Schölkopf, and A. Smola, "A kernel two-sample test," *J. Mach. Learn. Res.*, vol. 13, pp. 723–773, Mar. 2012.

[73] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter, "GANs trained by a two time-scale update rule converge to a local Nash equilibrium," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, 2017, pp. 6629–6640.

[74] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 2818–2826.

[75] H. Zhang, I. Goodfellow, D. Metaxas, and A. Odena. (2018). *Self-Attention Generative Adversarial Networks*. [Online]. Available: https://arxiv.org/abs/1805.08318

[76] L. Metz, B. Poole, D. Pfau, and J. Sohl-Dickstein, "Unrolled generative adversarial networks," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2017, pp. 1–25.

[77] N. Duan, D. Tang, P. Chen, and M. Zhou, "Question generation for question answering," in *Proc. Conf. Empir. Methods Nat. Lang. Process. (EMNLP)*, 2017, pp. 866–874.

Chaoyue Wang received the bachelor's degree from Tianjin University, Tianjin, China, and the Ph.D. degree from the University of Technology Sydney, Ultimo, NSW, Australia.

He is a Research Associate of machine learning and computer vision with the School of Computer Science, University of Sydney, Darlington, NSW, Australia. His current research interests include machine learning, deep learning, and generative models.

Dr. Wang was a recipient of the Distinguished Student Paper Award in the International Joint Conference on Artificial Intelligence in 2017.

Chang Xu received the Bachelor of Engineering degree from Tianjin University, Tianjin, China, and the Ph.D. degree from Peking University, Beijing, China.

He is a Lecturer of machine learning and computer vision with the School of Computer Science, University of Sydney, Darlington, NSW, Australia. He received fellowships from IBM, Armonk, NY, USA, and Baidu, Beijing. His current research interests include machine learning, data mining algorithms, and related applications in artificial intelligence and computer vision, including multiview learning, multilabel learning, visual search, and face recognition. He has published in prestigious journals and top-tier conferences in the above areas.

Xin Yao (F'03) received the B.Sc. degree from the University of Science and Technology of China (USTC), Hefei, China, in 1982, the M.Sc. degree from the North China Institute of Computing Technologies, Beijing, China, in 1985, and the Ph.D. degree from USTC in 1990.

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

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

Dacheng Tao (F'15) received the B.Eng. degree from the University of Science and Technology of China, Hefei, China, the M.Phil. degree from the Chinese University of Hong Kong, Hong Kong, and the Ph.D. degree from Birkbeck, University of London, London, U.K.

He is a Professor of computer science and an ARC Laureate Fellow with the School of Computer Science and the Faculty of Engineering and Information Technologies, and the Inaugural Director of the UBTECH Sydney Artificial Intelligence Centre, University of Sydney, Darlington, NSW, Australia. He mainly applies statistics and mathematics to artificial intelligence and data science. His research results have expounded in one monograph and over 200 publications in prestigious journals and prominent conferences, such as the IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, the IEEE TRANSACTIONS ON IMAGE PROCESSING, the IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS, the *International Journal of Computer Vision*, the *Journal of Machine Learning Research*, Neural Information Processing Systems, International Conference on Machine Learning, IEEE Conference on Computer Vision and Pattern Recognition, IEEE International Conference on Computer Vision, European Conference on Computer Vision, IEEE International Conference on Data Mining, Association for Computing Machinery's Special Interest Group on Knowledge Discovery and Data Mining.

Mr. Tao was a recipient of several best paper awards, such as the Best Theory/Algorithm Paper Runner Up Award in IEEE ICDM'07, the Best Student Paper Award in IEEE ICDM'13, the Distinguished Paper Award in the 2018 IJCAI, the 2014 ICDM 10-Year Highest-Impact Paper Award, and the 2017 IEEE Signal Processing Society Best Paper Award, the 2015 Australian Scopus-Eureka Prize, and the 2018 IEEE ICDM Research Contributions Award. He is a fellow of the Australian Academy of Science, American Association for the Advancement of Science, International Association for Pattern Recognition, Optical Society of America, Society of Photo-Optical Instrumentation Engineers.