Evolving GAN Formulations for Higher Quality Image Synthesis

Santiago Gonzalez[1,2,*], Mohak Kant[2], Risto Miikkulainen[1,2]
[1]The University of Texas at Austin, [2]Cognizant AI Labs

Abstract

Generative Adversarial Networks (GANs) have extended deep learning to complex generation and translation tasks across different data modalities. However, GANs are notoriously difficult to train: Mode collapse and other instabilities in the training process often degrade the quality of the generated results, such as images. This paper presents a new technique called TaylorGAN for improving GANs by discovering customized loss functions for each of its two networks. The loss functions are parameterized as Taylor expansions and optimized through multiobjective evolution. On an image-to-image translation benchmark task, this approach qualitatively improves generated image quality and quantitatively improves two independent GAN performance metrics. It therefore forms a promising approach for applying GANs to more challenging tasks in the future.

1 Introduction

Generative Adversarial Networks (GANs) have recently emerged as a promising technique for building models that generate new samples according to a distribution within a dataset. In GANs, two separate networks—a generator and a discriminator—are trained in tandem in an adversarial fashion: The generator attempts to synthesize samples that the discriminator believes is real, while the discriminator attempts to differentiate between samples from the generator and samples from a ground-truth dataset. However, GANs are challenging to train. Training often suffers from instabilities that can lead to low-quality and potentially low-variety generated samples. These difficulties have lead many researchers to try formulating better GANs, primarily by designing new generator and discriminator loss functions by hand.

Neuroevolution may potentially offer a solution to this problem. It has recently been extended from optimizing network weights and topologies to designing deep learning architectures (Stanley et al., 2019; Real et al., 2019; Liang et al., 2019b). Advances in this field,—known as evolutionary metalearning—have resulted in designs that outperform those that are manually-tuned. One particular family of techniques—loss-function metalearning—has allowed for neural networks to be trained more quickly, with higher accuracy, and better robustness (Gonzalez and Miikkulainen, 2020, 2021). Perhaps loss-function metalearning can be adapted to improve GANs?

In this paper, such a technique is developed to evolve entirely new GAN formulations that outperform the standard Wasserstein loss. Leveraging the TaylorGLO loss-function parameterization approach (Gonzalez and Miikkulainen, 2021), separate loss functions are constructed for the two GAN networks. A genetic algorithm is then used to optimize their parameters against two non-differentiable objectives. A composite transformation of these objectives (Shahrzad et al., 2018) is further used to enhance the multiobjective search.

*Current affiliation: Apple Inc.

This TaylorGAN approach is evaluated experimentally in an image-to-image translation benchmark task where the goal is to generate photorealistic building images based on a building segment map. The CMP Facade dataset (Tyleček and Šára, 2013) is used as the training data and the pix2pix-HD conditional GAN (Wang et al., 2018) as the generative model. The approach is found to both qualitatively enhance generated image quality and quantitatively improve the two metrics. The evaluation thus demonstrates how evolution can improve a leading conditional GAN design by replacing manually designed loss functions with those optimized by a multiobjective genetic algorithm.

Section 2 reviews key literature in GANs, motivating the evolution of their loss functions. The next section describes the TaylorGLO metalearning technique for optimizing loss-functions in general. Section 4 introduces the TaylorGAN variation of it, focusing on how the TaylorGLO loss-function parameterization is leveraged for evolving GANs. Section 5 details the experimental configuration and evaluation methodologies. In Section 6, TaylorGAN's efficacy is evaluated on the benchmark task. Section 7 places these these findings in the general context of the GAN literature and describes potential avenues for future work.

2 Variations of GAN Architectures

Generative Adversarial Networks (GANs; Goodfellow et al., 2014), are a type of generative model consisting of a pair of networks, a generator and discriminator, that are trained in tandem. GANs are a modern successor to Variational Autoencoders (VAEs; Kingma and Welling, 2014) and Boltzmann Machines (Hinton and Sejnowski, 1983), including Restricted Boltzmann Machines (Smolensky, 1986) and Deep Boltzmann Machines (Salakhutdinov and Hinton, 2009).

The following subsections review prominent GAN methods. Key GAN formulations, and the relationships between them, are described. Consistent notation (shown in Table 1) is used, consolidating the extensive variety of notation in the field.

Table 1: GAN Notation Decoder

Symbol	**Description**
$G(\boldsymbol{x}, \theta_G)$	Generator function
$D(\boldsymbol{z}, \theta_D)$	Discriminator function
$\mathbb{P}_{\text{data}}$	Probability distribution of the original data
$\mathbb{P}_z$	Latent vector noise distribution
$\mathbb{P}_g$	Probability distribution of $G(\boldsymbol{z})$
$\boldsymbol{x}$	Data, where $\boldsymbol{x} \sim \mathbb{P}_{\text{data}}$
$\tilde{\boldsymbol{x}}$	Generated data
$\boldsymbol{z}$	Latent vector, where $\boldsymbol{z} \sim \mathbb{P}_z$
$\boldsymbol{c}$	Condition vector
λ	Various types of weights / hyperparameters

2.1 Overview

A GAN's generator and discriminator are set to compete with each other in a minimax game, attempting to reach a Nash equilibrium (Nash, 1951; Heusel et al., 2017). Throughout the training process, the generator aims to transform samples from a prior noise distribution into data, such as an image, that tricks the discriminator into thinking it has been sampled from the real data's distribution. Simultaneously, the discriminator aims to determine whether a given sample came from the real data's distribution, or was generated from noise.

Unfortunately, GANs are difficult to train, frequently exhibiting instability, i.e., mode collapse, where all modes of the target data distribution are not fully represented by the generator (Radford et al., 2015; Metz et al., 2016; Isola et al., 2016; Mao et al., 2017; Arjovsky et al., 2017; Gulrajani et al., 2017; Mao et al., 2018). GANs that operate on image data often suffer from visual artifacts and blurring of generated images (Isola et al., 2016; Odena et al., 2016). Additionally, datasets with low variability have been found to degrade GAN performance (Mao et al., 2018).

GANs are also difficult to evaluate quantitatively, typically relying on metrics that attempt to embody vague notions of quality. Popular GAN image scoring metrics, for example, have been found to have many pitfalls, including cases where two samples of clearly disparate quality may have similar values (Borji, 2019).

2.2 Original Minimax and Non-Saturating GAN

Using the notation described in Table 1, the original minimax GAN formulation by Goodfellow et al. (2014) can be defined as

$$\min_{\theta_G} \max_{\theta_D} \; \mathbb{E}_{\boldsymbol{x}\sim\mathbb{P}_{\text{data}}} \left[\log D(\boldsymbol{x})\right] + \mathbb{E}_{\boldsymbol{z}\sim\mathbb{P}_z} \left[\log\left(1 - D(G(\boldsymbol{z}))\right)\right] \; . \tag{1}$$

This formulation can be broken down into two separate loss functions, one each for the discriminator and generator:

$$\mathcal{L}_D \;=\; -\frac{1}{n}\sum_{i=1}^{n} \left[\log D(x_i) + \log(1 - D(G(z_i)))\right] \text{ , and} \tag{2}$$

$$\mathcal{L}_G \;=\; \frac{1}{n}\sum_{i=1}^{n} \log(1 - D(G(z_i))) \; . \tag{3}$$

The discriminator's loss function is equivalent to a sigmoid cross-entropy loss when thought of as a binary classifier. Goodfellow et al. (2014) proved that training a GAN with this formulation is equivalent to minimizing the Jensen-Shannon divergence between $\mathbb{P}_g$ and $\mathbb{P}_{\text{data}}$, i.e. a symmetric divergence metric based on the Kullback-Leibler divergence.

In the above formulation the generator's loss saturates quickly since the discriminator learns to reject the novice generator's samples early on in training. To resolve this problem, Goodfellow et al. provided a second "non-saturating" formulation with the same fixed-point dynamics, but better, more intense gradients for the generator early on:

$$\max_{\theta_D} \quad \mathbb{E}_{\boldsymbol{x}\sim\mathbb{P}_{\text{data}}} \left[\log D(\boldsymbol{x})\right] + \mathbb{E}_{\boldsymbol{z}\sim\mathbb{P}_z} \left[\log\left(1 - D(G(\boldsymbol{z}))\right)\right] \; , \tag{4}$$

$$\max_{\theta_G} \quad \mathbb{E}_{\boldsymbol{z}\sim\mathbb{P}_z} \left[\log D(G(\boldsymbol{z}))\right] \; . \tag{5}$$

Each GAN training step consists of training the discriminator for k steps, while sequentially training the generator for only one step. This difference in steps for both networks helps prevent the discriminator from learning too quickly and overpowering the generator.

Alternatively, Unrolled GANs (Metz et al., 2016) aimed to prevent the discriminator from overpowering the generator by using a discriminator which has been unrolled for a certain number of steps in the generator's loss, thus allowing the generator to train against a more optimal discriminator. More recent GAN work instead uses a two time-scale update rule (TTUR; Heusel et al., 2017), where the two networks are trained under different learning rates for one step each. This approach has proven to converge more reliably to more desirable solutions.

Unfortunately, with both minimax and non-saturating GANs the generator gradients vanish for samples that are on the correct side of the decision boundary but far from the true data distribution (Mao et al., 2017, 2018). The Wasserstein GAN, described next, is designed to solve this problem.

2.3 Wasserstein GAN

The Wasserstein GAN (WGAN; Arjovsky et al., 2017) is arguably one of the most impactful developments in the GAN literature since the original formulation by Goodfellow et al. (2014). WGANs minimize the Wasserstein-1 distance between $\mathbb{P}_g$ and $\mathbb{P}_{\text{data}}$, rather than the Jensen-Shannon divergence, in an attempt to avoid vanishing gradient and mode collapse issues. In the context of GANs, the Wasserstein-1 distance can be defined as

$$W(\mathbb{P}_g, \mathbb{P}_{\text{data}}) = \inf_{\gamma \in \Pi(\mathbb{P}_g, \mathbb{P}_{\text{data}})} \mathbb{E}_{(\boldsymbol{u},\boldsymbol{v})\sim\gamma} \left[\|\boldsymbol{u} - \boldsymbol{v}\|\right] \ , \tag{6}$$

where, $\gamma(\boldsymbol{u}, \boldsymbol{v})$ represents the amount of mass that needs to move from $\boldsymbol{u}$ to $\boldsymbol{v}$ for $\mathbb{P}_g$ to become $\mathbb{P}_{\text{data}}$. This formulation with the infimum is intractable, but the Kantorovich-Rubinstein duality (Villani, 2009) with a supremum makes the Wasserstein-1 distance tractable, while imposing a 1-Lipschitz smoothness constraint:

$$W(\mathbb{P}_g, \mathbb{P}_{\text{data}}) = \sup_{\|f\|_L \leq 1} \mathbb{E}_{\boldsymbol{u}\sim\mathbb{P}_g} \left[f(\boldsymbol{u})\right] - \mathbb{E}_{\boldsymbol{u}\sim\mathbb{P}_{\text{data}}} \left[f(\boldsymbol{u})\right] \ , \tag{7}$$

which translates to the training objective

$$\min_{\theta_G} \max_{\theta_D \in \Theta_D} \mathbb{E}_{\boldsymbol{x}\sim\mathbb{P}_{\text{data}}} \left[D(\boldsymbol{x})\right] - \mathbb{E}_{\boldsymbol{z}\sim\mathbb{P}_z} \left[D(G(\boldsymbol{z}))\right] \ , \tag{8}$$

where Θ_D is the set of all parameters for which D is a 1-Lipschitz function.

WGANs are an excellent example of how generator and discriminator loss functions can profoundly impact the quality of generated samples and the prevalence of mode collapse. However, the WGAN has a 1-Lipschitz constraint that needs to be maintained throughout training for the formulation to work. WGANs enforce the constraint via gradient clipping, at the cost of requiring an optimizer that does not use momentum, i.e., RMSProp (Tieleman and Hinton, 2012) rather than Adam (Kingma and Ba, 2015).

To resolve the issues caused by gradient clipping, a subsequent formulation, WGAN-GP (Gulrajani et al., 2017), added a gradient penalty regularization term to the discriminator loss:

$$GP = \lambda \, \mathbb{E}_{\hat{\boldsymbol{x}}\sim\mathbb{P}_{\hat{x}}} \left[\left(\|\nabla_{\hat{\boldsymbol{x}}} D(\hat{\boldsymbol{x}})\|_2 - 1\right)^2\right] \ , \tag{9}$$

where $\mathbb{P}_{\hat{x}}$ samples uniformly along lines between $\mathbb{P}_{\text{data}}$ and $\mathbb{P}_g$. The gradient penalty enforces a soft Lipschitz smoothness constraint, leading to a more stationary loss surface than when gradient clipping is used, which in turn makes it possible to use momentum-based optimizers. The gradient penalty term has even been successfully used in non-Wasserstein GANs (Fedus et al., 2018; Mao et al., 2018). However, gradient penalties can increase memory and compute costs (Mao et al., 2018).

2.4 Least-Squares GAN

Another attempt to solve the issue of vanishing gradients is the Least-Squares GAN (LSGAN Mao et al., 2017). It defines the training objective as

$$\min_{\theta_D} \quad \frac{1}{2}\,\mathbb{E}_{\boldsymbol{x}\sim\mathbb{P}_{\text{data}}}\left[\left(D(\boldsymbol{x})-b\right)^2\right]+\mathbb{E}_{\boldsymbol{z}\sim\mathbb{P}_z}\left[\left(D(G(\boldsymbol{z}))-a\right)^2\right]\,, \tag{10}$$

$$\min_{\theta_G} \quad \frac{1}{2}\,\mathbb{E}_{\boldsymbol{z}\sim\mathbb{P}_z}\left[\left(D(G(\boldsymbol{z}))-c\right)^2\right]\,, \tag{11}$$

where a is the label for generated data, b is the label for real data, and c is the label that G wants to trick D into believing for generated data. In practice, typically $a=0, b=1, c=1$. However, subsequently, $a=-1, b=-1, c=0$ were found to result in faster convergence, making it the recommended parameter setting (Gulrajani et al., 2017). Training an LSGAN was shown to be equivalent to minimizing the Pearson χ^2 divergence (Pearson, 1900) between $\mathbb{P}_{\text{data}}+\mathbb{P}_g$ and $2*\mathbb{P}_g$. Generated data quality can oscillate throughout the training process (Mao et al., 2018), indicating a disparity between data quality and loss.

2.5 Conditional GAN

Traditional GANs learn how to generate data from a latent space, i.e. an embedded representation of the training data that the generator constructs. Typically, the elements of a latent space have no immediately intuitive meaning (Chen et al., 2016; Larsen et al., 2015). Thus, GANs can generate novel data, but there is no way to steer the generation process to generate particular types of data. For example, a GAN that generates images of human faces cannot be explicitly told to generate a face with a particular hair color or of a specific gender. While techniques have been developed to analyze this latent space (Volz et al., 2018; Li et al., 2019), or build more interpretable latent spaces during the training process (Chen et al., 2016), they do not necessarily translate a human's prior intuition correctly or make use of labels when they are available. To tackle this problem, Conditional GANs, first proposed as future work by Goodfellow et al. (2014) and subsequently developed Mirza and Osindero (2014), allow directly targetable features (i.e., conditions) to be an integral part of the generator's input.

The conditioned training objective for a minimax GAN can be defined, without loss of generality, as

$$\min_{\theta_G}\max_{\theta_D}\;\mathbb{E}_{\boldsymbol{x}\sim\mathbb{P}_{\text{data}}}\left[\log D(\boldsymbol{x}\oplus\boldsymbol{c})\right]+\mathbb{E}_{\boldsymbol{z}\sim\mathbb{P}_z}\left[\log\left(1-D(G(\boldsymbol{z}\oplus\boldsymbol{c}))\right)\right]\,, \tag{12}$$

where $\boldsymbol{z}\oplus\boldsymbol{c}$ is basic concatenation of vectors. During training, the condition vector, $\boldsymbol{c}$, arises from the sampling process that produces each $\boldsymbol{x}$. This same framework can be used to design conditioned variants of other GAN formulations.

Conditional GANs have enjoyed great successes as a result of their flexibility, even in the face of large, complex condition vectors, which may even be whole images. They enable new applications for GANs, including repairing software vulnerabilities (framed as sequence to sequence translation; Harer

et al., 2018), integrated circuit mask design (Alawieh et al., 2019), and image to image translation (Isola et al., 2016)—the generation of images given text (Reed et al., 2016)—which is used as the target setting for this paper. Notably, conditional GANs can increase the quality of generated samples for labeled datasets, even when conditioned generation is not needed (van den Oord et al., 2016). Conditional GANs are therefore used as the platform for the TaylorGAN technique described in the next section.

2.6 Opportunity: Optimizing Loss Functions

The GAN formulations described above all have one property in common: The generator and discriminator loss functions have been arduously derived by hand. A GAN's performance and stability is greatly impacted by the choice of loss functions. Different regularization terms, such as the aforementioned gradient penalty can also affect a GAN's training. These elements of the GAN are typically designed to minimize a specific divergence. However, a GAN does not need to decrease a divergence at every step in order to reach the Nash equilibrium (Fedus et al., 2018). In this situation, an automatic loss-function optimization system may find novel GAN loss functions with more desirable properties. Such a system is presented in Section 4 and evaluated on conditional GANs in Section 6. The basic method for evolving loss functions, TaylorGLO, is reviewed in the next section.

3 Evolution of Loss Functions

Loss-function metalearning makes it possible to regularize networks automatically; TaylorGLO is a flexible and scalable implementation of this idea based on multivariate Taylor expansions.

3.1 Motivation

Loss-function metalearning for deep networks was first introduced by Gonzalez and Miikkulainen (2020) as an automatic way to find customized loss functions that optimize a performance metric for a model. The technique, a genetic programming approach named GLO, discovered one particular loss function, Baikal, that improves classification accuracy, training speed, and data utilization. Intuitively, Baikal achieved these properties through a form of regularization that ensured the model would not become overly confident in its predictions. That is, instead of monotonically decreasing the loss when the output gets closer to the correct value, Baikal loss increases rapidly when the output is almost correct, thus discouraging extreme accuracy.

TaylorGLO (Gonzalez and Miikkulainen, 2021) is a scalable reformulation of the GLO approach.[1] Instead of trees evolved through genetic programming, TaylorGLO represents loss functions as parameterizations of multivariate Taylor polynomials. It is then possible to evolve the parameters directly with CMA-ES, which makes it possible to scale to models with millions of trainable parameters and a variety of deep learning architectures.

[1] Open-source code for TaylorGLO is available at `https://github.com/cognizant-ai-labs/taylorglo`.

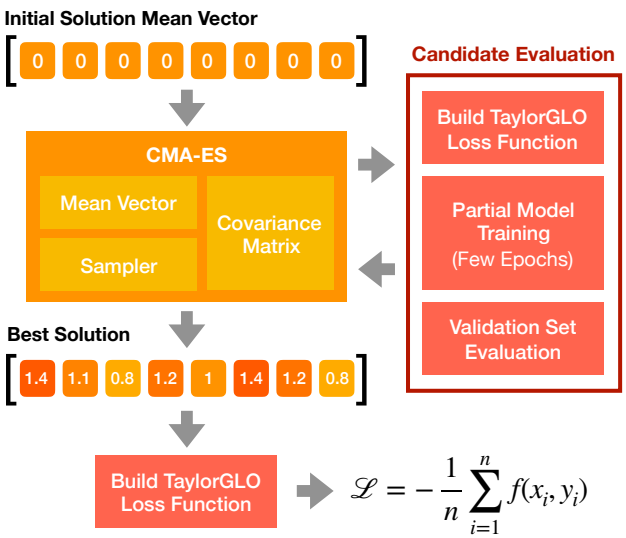

Figure 1: The TaylorGLO method (Gonzalez and Miikkulainen, 2021). Loss functions are represented by fixed-size vectors whose elements parameterize modified Taylor polynomials. Starting with a population of initially unbiased loss functions (i.e., vectors around the origin), CMA-ES optimizes their Taylor expansion parameters in order to maximize validation accuracy after partial training. The candidate with the highest accuracy is chosen as the final, best solution. This approach biases the search towards functions with useful properties, and is also amenable to theoretical analysis, as shown in this paper.

3.2 Loss Functions as Multivariate Taylor Expansions

Taylor expansions (Taylor, 1715) represent differentiable functions within the neighborhood of a point using a polynomial series. In the univariate case, given a $C^{k_{\max}}$ smooth (i.e., first through $k_{\max}$ derivatives are continuous), real-valued function, $f(x) : \mathbb{R} \to \mathbb{R}$, a kth-order Taylor approximation at point $a \in \mathbb{R}$, $\hat{f}_k(x, a)$, where $0 \leq k \leq k_{\max}$, can be constructed as

$$\hat{f}_k(x, a) = \sum_{n=0}^{k} \frac{1}{n!} f^{(n)}(a)(x - a)^n. \tag{13}$$

This formulation can be extended to the multivariate case by defining an nth-degree multi-index, $\alpha = (\alpha_1, \alpha_2, \ldots, \alpha_n)$, where $\alpha_i \in \mathbb{N}_0$, $|\alpha| = \sum_{i=1}^{n} \alpha_i$, $\alpha! = \prod_{i=1}^{n} \alpha_i!$. $\mathbf{x}^\alpha = \prod_{i=1}^{n} x_i^{\alpha_i}$, and $\mathbf{x} \in \mathbb{R}^n$. Multivariate partial derivatives can be concisely written using a multi-index as

$$\partial^\alpha f = \partial_1^{\alpha_1} \partial_2^{\alpha_2} \cdots \partial_n^{\alpha_n} f = \frac{\partial^{|\alpha|}}{\partial x_1^{\alpha_1} \partial x_2^{\alpha_2} \cdots \partial x_n^{\alpha_n}}. \tag{14}$$

Thus, discounting the remainder term, the multivariate Taylor expansion for $f(\mathbf{x})$ at $\mathbf{a}$ is

$$\hat{f}_k(\mathbf{x}, \mathbf{a}) = \sum_{\forall \alpha, |\alpha| \leq k} \frac{1}{\alpha!} \partial^\alpha f(\mathbf{a})(\mathbf{x} - \mathbf{a})^\alpha. \tag{15}$$

The unique partial derivatives in $\hat{f}_k$ and $\mathbf{a}$ are parameters for a kth order Taylor expansion. Thus, a kth order Taylor expansion of a function in n variables requires n parameters to define the center, $\mathbf{a}$,

and one parameter for each unique multi-index α, where $|\alpha| \leq k$. That is:

$$\#_{\text{parameters}}(n,k) = n + \binom{n+k}{k} = n + \frac{(n+k)!}{n!\,k!} \ . \tag{16}$$

The multivariate Taylor expansion can be leveraged for loss-function parameterization (Gonzalez and Miikkulainen, 2021). Let an n-class classification loss function be defined as $\mathcal{L}_{\text{Log}} = -\frac{1}{n}\sum_{i=1}^{n} f(x_i, y_i)$. The function $f(x_i, y_i)$ can be replaced by its kth-order, bivariate Taylor expansion, $\hat{f}_k(x, y, a_x, a_y)$. For example, a loss function in $\mathbf{x}$ and $\mathbf{y}$ has the following third-order parameterization with parameters θ (where $\mathbf{a} = \langle \theta_0, \theta_1 \rangle$):

$$\begin{aligned}\mathcal{L}(\mathbf{x},\mathbf{y}) = -\frac{1}{n}\sum_{i=1}^{n}\Big[&\theta_2 + \theta_3(y_i - \theta_1) + \tfrac{1}{2}\theta_4(y_i - \theta_1)^2 \\ &+\tfrac{1}{6}\theta_5(y_i - \theta_1)^3 + \theta_6(x_i - \theta_0) + \theta_7(x_i - \theta_0)(y_i - \theta_1) \\ &+\tfrac{1}{2}\theta_8(x_i - \theta_0)(y_i - \theta_1)^2 + \tfrac{1}{2}\theta_9(x_i - \theta_0)^2 \\ &+\tfrac{1}{2}\theta_{10}(x_i - \theta_0)^2(y_i - \theta_1) + \tfrac{1}{6}\theta_{11}(x_i - \theta_0)^3\Big]\end{aligned} \tag{17}$$

As was shown by Gonzalez and Miikkulainen (2021), the technique makes it possible to train neural networks that are more accurate and learn faster than those with tree-based loss function representations. Representing loss functions in this manner guarantees that the functions are smooth, do not have poles, can be implemented through addition and multiplication, and can be trivially differentiated. The search space is locally smooth and has a tunable complexity parameter (the order of expansion), making it possible to find valid loss functions consistently and with high frequency. These properties are not necessarily held by alternative function approximators, such as Fourier expansions, Padé approximants, Laurent polynomials, and Polyharmonic splines (Gonzalez and Miikkulainen, 2021).

3.3 The TaylorGLO Method

TaylorGLO (Figure 1) aims to find the optimal parameters for a loss function represented as a multivariate Taylor expansion. The parameters for a Taylor approximation (i.e., the center point and partial derivatives) are referred to as $\theta_{\hat{f}}$: $\theta_{\hat{f}} \in \Theta$, $\Theta = \mathbb{R}^{\#_{\text{parameters}}}$. TaylorGLO strives to find the vector $\theta^*_{\hat{f}}$ that parameterizes the optimal loss function for a task. Because the values are continuous, as opposed to discrete graphs of the original GLO, it is possible to use continuous optimization methods.

In particular, Covariance Matrix Adaptation Evolutionary Strategy (CMA-ES; Hansen and Ostermeier, 1996) is a popular population-based, black-box optimization technique for rugged, continuous spaces. CMA-ES functions by maintaining a covariance matrix around a mean point that represents a distribution of solutions. At each generation, CMA-ES adapts the distribution to better fit evaluated objective values from sampled individuals. In this manner, the area in the search space that is being sampled at each step grows, shrinks, and moves dynamically as needed to maximize sampled candidates' fitnesses. TaylorGLO uses the $(\mu/\mu, \lambda)$ variant of CMA-ES (Hansen and Ostermeier, 2001), which incorporates weighted rank-μ updates (Hansen and Kern, 2004) to reduce the number of objective function evaluations needed.

In order to find $\theta^*_{\hat{f}}$, at each generation CMA-ES samples points in Θ. Their fitness is determined by training a model with the corresponding loss function and evaluating the model on a validation

dataset. Fitness evaluations may be distributed across multiple machines in parallel and retried a limited number of times upon failure. An initial vector of $\theta_{\hat{f}} = \mathbf{0}$ is chosen as a starting point in the search space to avoid bias.

Fully training a model can be prohibitively expensive in many problems. However, performance near the beginning of training is usually correlated with performance at the end of training, and therefore it is enough to train the models only partially to identify the most promising candidates. This type of approximate evaluation is common in metalearning (Grefenstette and Fitzpatrick, 1985; Jin, 2011). An additional positive effect is that evaluation then favors loss functions that learn more quickly.

For a loss function to be useful, it must have a derivative that depends on the prediction. Therefore, internal terms that do not contribute to $\frac{\partial}{\partial \mathbf{y}} \mathcal{L}_f(\mathbf{x}, \mathbf{y})$ can be trimmed away. This step implies that any term t within $f(x_i, y_i)$ with $\frac{\partial}{\partial y_i} t = 0$ can be replaced with 0. For example, this refinement simplifies Equation 17, providing a reduction in the number of parameters from twelve to eight:

$$\begin{aligned} \mathcal{L}(\mathbf{x}, \mathbf{y}) = -\frac{1}{n} \sum_{i=1}^{n} \Big[& \theta_2(y_i - \theta_1) + \tfrac{1}{2}\theta_3(y_i - \theta_1)^2 + \tfrac{1}{6}\theta_4(y_i - \theta_1)^3 \\ & + \theta_5(x_i - \theta_0)(y_i - \theta_1) + \tfrac{1}{2}\theta_6(x_i - \theta_0)(y_i - \theta_1)^2 \\ & + \tfrac{1}{2}\theta_7(x_i - \theta_0)^2(y_i - \theta_1) \Big] . \end{aligned} \tag{18}$$

Building on this foundation, the method for evolving GAN formulations is described next.

4 The TaylorGAN Approach

As GANs have grown in popularity, the difficulties involved in training them have become increasingly evident. The loss functions used to train a GAN's generator and discriminator constitute the core of how GANs are formulated. Thus, optimizing these loss functions jointly can result in better GANs. This section presents an extension of TaylorGLO to evolve loss functions for GANs. Images generated in this way improve both visually and quantitatively, as the experiments in Section 6 show.

TaylorGLO parameterization represents a loss function as a modified third-degree Taylor polynomial. Such a parameterization has many desirable properties, such as smoothness and continuity, that make it amenable for evolution (Gonzalez and Miikkulainen, 2021). In TaylorGAN, there are three functions that need to be optimized jointly (using the notation described in Table 1):

1. The component of the discriminator's loss that is a function of $D(\boldsymbol{x})$, the discriminator's output for a real sample from the dataset,
2. The synthetic / fake component of the discriminator's loss that is a function of $D(G(\boldsymbol{z}))$, the discriminator's output from the generator that samples $\boldsymbol{z}$ from the latent distribution), and
3. The generator's loss, a function of $D(G(\boldsymbol{z}))$.

The discriminator's full loss is simply the sum of components (1) and (2). Table 2 shows how existing GAN formulations can be broken down into this tripartite loss.

These three functions can be evolved jointly. GAN loss functions have a single input, i.e. $D(\boldsymbol{x})$ or $D(G(\boldsymbol{z}))$. Thus, a set of three third-order TaylorGLO loss functions for GANs requires only 12 parameters to be optimized, making the technique quite efficient.

Table 2: **Interpretation of existing GAN formulations.** These three components are all that is needed to define the discriminator's and generator's loss functions (sans regularization terms). Thus, TaylorGAN can discover and optimize new GAN formulations by jointly evolving three separate functions.

Formulation	**Loss D (real)** $\mathbb{E}_{\boldsymbol{x}\sim\mathbb{P}_{\text{data}}}$	**Loss D (fake)** $\mathbb{E}_{\boldsymbol{z}\sim\mathbb{P}_z}$	**Loss G (fake)** $\mathbb{E}_{\boldsymbol{z}\sim\mathbb{P}_z}$
GAN (minimax; Goodfellow et al., 2014)	$-\log D(\boldsymbol{x})$	$-\log(1-D(G(\boldsymbol{z})))$	$\log(1-D(G(\boldsymbol{z})))$
GAN (non-saturating; Goodfellow et al., 2014)	$-\log D(\boldsymbol{x})$	$-\log(1-D(G(\boldsymbol{z})))$	$-\log D(G(\boldsymbol{z}))$
WGAN (Arjovsky et al., 2017)	$-D(\boldsymbol{x})$	$D(G(\boldsymbol{z}))$	$-D(G(\boldsymbol{z}))$
LSGAN (Mao et al., 2017)	$\frac{1}{2}(D(\boldsymbol{x})-1)^2$	$\frac{1}{2}(D(G(\boldsymbol{z})))^2$	$\frac{1}{2}(D(G(\boldsymbol{z}))-1)^2$

Fitness for each set of three functions requires a different interpretation than in regular TaylorGLO. Since GANs cannot be thought of as having an accuracy, a different metric needs to be used. The choice of fitness metric depends on the type of problem and target application. In the uncommon case where the training data's sampling distribution is known, the clear choice is the divergence between such a distribution and the distribution of samples from the generator. This approach will be used in the experiments below.

Reliable metrics of visual quality are difficult to define. Individual image quality metrics can be exploited by adversarially constructed, lesser-quality images (Borji, 2019). For this reason, TaylorGAN utilizes a combination of two or more metrics, and multiobjective optimization of them. Good solutions are usually located near the middle of the resulting Pareto front, and they can be found effectively through an objective transformation technique called Composite Objectives (Shahrzad et al., 2018). In this technique, evolution is performed against a weighted sum of metrics. Individual metrics are scaled such that their ranges of typical values match. Thus, if one metric improves, overall fitness will only increases if there is not a comparable regression along another metric.

5 Experimental Setup

The technique was integrated into the LEAF evolutionary AutoML framework (Liang et al., 2019a). TaylorGAN parameters were evolved by the LEAF genetic algorithm as if they were hyperparameters. The implementation of CoDeepNEAT (Miikkulainen et al., 2019) for neural architecture search in LEAF was not used.

The technique was evaluated on the CMP Facade (Tyleček and Šára, 2013) dataset with a pix2pix-HD model (Wang et al., 2018). The dataset consists of only 606 perspective-corrected 256×256 pixel images of building facades. Each image has a corresponding annotation image that segments facades into twelve different components, such as windows and doors. The objective is for the model to take an arbitrary annotation image as an input, and generate a photorealistic facade as output. The dataset was split into a training set with 80% of the images, and validation and testing sets, each with a disjoint 10% of the images.

Two metrics were used to evaluate loss function candidates: (1) structural similarity index measure (SSIM; Wang et al., 2004) between generated and ground-truth images, and (2) perceptual distance, implemented as the L_1 distance between VGG-16 (Simonyan and Zisserman, 2014) ImageNet

(Russakovsky et al., 2015) embeddings for generated and ground-truth images. During evolution, a composite objective (Shahrzad et al., 2018) of these two metrics was used to evaluate candidates. The metrics were normalized (i.e., SSIM was multiplied by 17 and perceptual distance by -1) to have a similar impact on evolution.

The target GAN model, pix2pix-HD, is a refinement of the seminal pix2pix model (Isola et al., 2016). Both models generate images conditioned upon an input image. Thus, they are trained with paired images. The baseline was trained with the Wasserstein loss (Arjovsky et al., 2017) and spectral normalization (Miyato et al., 2018) to enforce the Lipschitz constraint on the discriminator. The pix2pix-HD model is also trained with additive perceptual distance and discriminator feature losses. Both additive losses are multiplied by ten in the baseline. Models were trained for 60 epochs.

When running experiments, each of the twelve TaylorGAN parameters was evolved within $[-10, 10]$. The learning rate and weights for both additive losses were also evolved since the baseline values, which are optimal for the Wasserstein loss, may not necessarily be optimal for TaylorGAN loss functions.

6 Results

TaylorGAN found a set of loss functions that outperformed the original Wasserstein loss with spectral normalization. After 49 generations of evolution, it discovered the loss functions

$$\begin{aligned}\mathcal{L}_{D_{\text{real}}} = 5.6484\ (D(\boldsymbol{x}) - 8.3399) + 9.4935\ (D(\boldsymbol{x}) - 8.3399)^2 \\ +8.2695\ (D(\boldsymbol{x}) - 8.3399)^3\end{aligned} \tag{19}$$

$$\begin{aligned}\mathcal{L}_{D_{\text{fake}}} = 6.7549\ (D(G(\boldsymbol{z})) - 8.6177) + 2.4328\ (D(G(\boldsymbol{z})) - 8.6177)^2 \\ +8.0006\ (D(G(\boldsymbol{z})) - 8.6177)^3\end{aligned} \tag{20}$$

$$\begin{aligned}\mathcal{L}_{G_{\text{fake}}} = 0.0000\ (D(G(\boldsymbol{z})) - 5.2232) + 5.2849\ (D(G(\boldsymbol{z})) - 5.2232)^2 \\ +0.0000\ (D(G(\boldsymbol{z})) - 5.2232)^3.\end{aligned} \tag{21}$$

A learning rate of 0.0001, discriminator feature loss weight of 4.0877, and perceptual distance loss weight of 10.3155 evolved for this candidate.

Figure 2 compares images for five random test samples that were generated with both the Wasserstein baseline and metalearned TaylorGAN loss functions. Visually, the TaylorGAN samples have more realistic coloration and details than the baseline. Baseline images all have an orange tint, while TaylorGAN images more closely match ground-truth images' typical coloration. Note that color information is not included in the model's input, so per-sample color matching is not possible. Additionally, TaylorGAN images tend to have higher-quality fine-grained details. For example, facade textures are unnaturally smooth and clean in the baseline, almost appearing to be made of plastic.

Quantitatively, the TaylorGAN model also outperforms the Wasserstein baseline. Across ten Wasserstein baseline runs, the average test-set SSIM was 9.4359 and the average test-set perceptual distance was 2129.5069. The TaylorGAN model improved both metrics, with a SSIM of 11.6615 and perceptual distance of 2040.2561.

Notably, the training set is very small, with fewer than 500 image pairs, showing how loss-function metalearning's benefits on small classification datasets also extend to GANs. Thus, metalearned loss functions are an effective way to train better GAN models, extending the types of problems to which evolutionary loss-function metalearning can be applied.

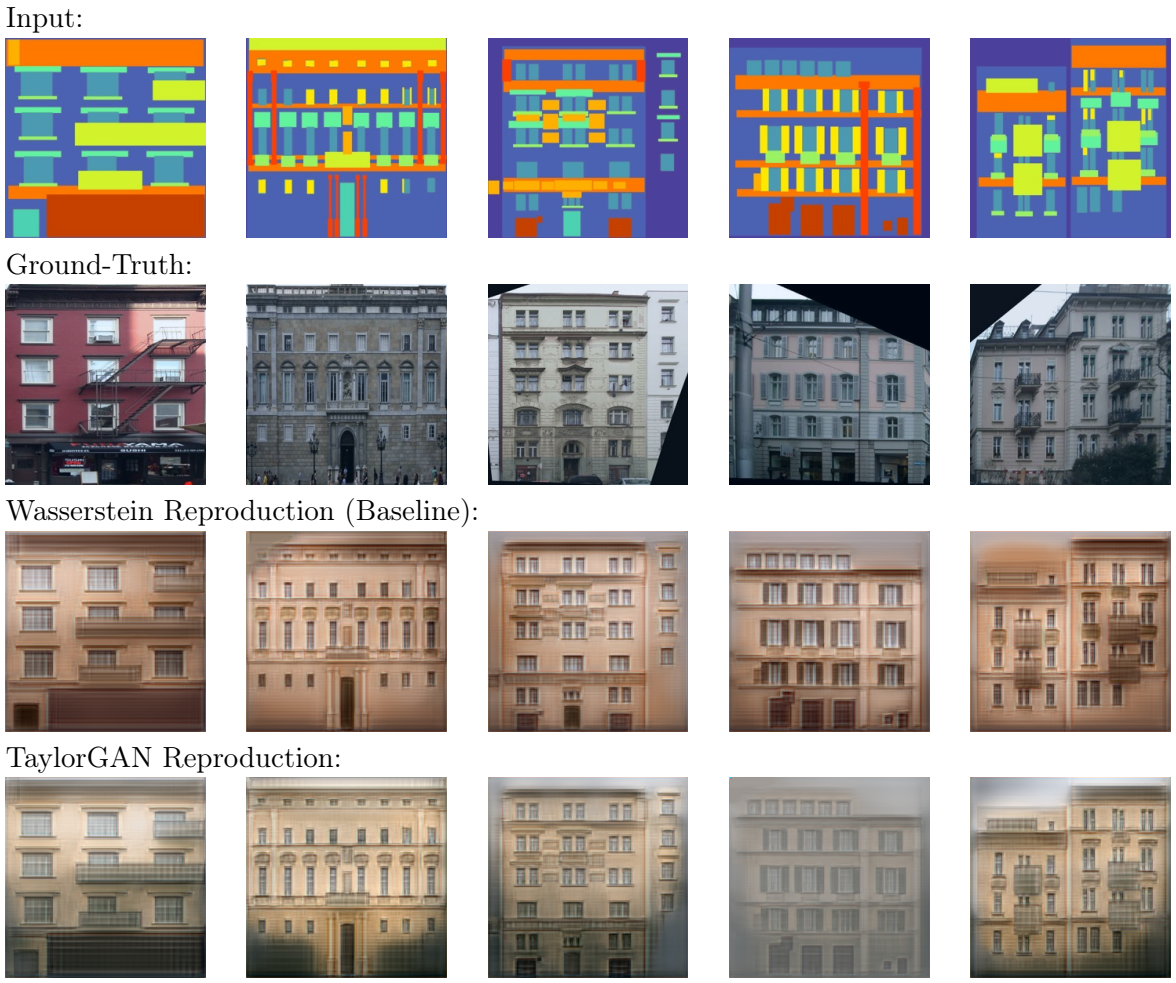

Figure 2: **Five random samples from the CMP Facade test dataset, comparing Wasserstein and TaylorGAN loss functions.** The loss functions are used to train pix2pix-HD models that take architectural element annotations (top row) and generate corresponding photorealistic images similar to the ground-truth (second row). Images from the model trained with TaylorGAN (bottom row) have a higher quality than the baseline (third row). TaylorGAN images have more realistic coloration, better separation of the buildings from the sky, and finer details than the baseline.

7 Discussion and future work

The results in this paper show that evolving GAN formulations is a promising direction for research. On the CMP Facade benchmark dataset, TaylorGAN discovered powerful loss functions: With them, GANs generated images that were qualitatively and quantitatively better than those produced by GANs with a Wasserstein loss. This unique application showcases the power and flexibility of evolutionary loss-function metalearning, and suggests that it may provide a crucial ingredient in making GANs more reliable and scalable to harder problems.

At first glance, optimizing GAN loss functions is difficult because it is difficult to quantify a GAN's performance. That is, performance can be improved on an individual metric without increasing the quality of generated images. Multiobjective evolution, via composite objectives, is thus a key technique that allows evolution to work on GAN formulations. That is, by optimizing against multiple metrics, each with their own negative biases, the effects of each individual metric's bias will not deleteriously affect the path evolution takes.

There are several avenues of future work with TaylorGAN. First, it can naturally be applied to different datasets and different types of GANs. While image-to-image translation is an important GAN domain, there are many others that can benefit from optimization, such as image super-resolution and unconditioned image generation. Since TaylorGAN customizes loss functions for a given task, dataset, and architecture, unique sets of loss functions could be discovered for each of them.

There is a wide space of metrics, such as Delta E perceptual color distance (Robertson, 1990), that quantify different aspects of image quality. They can be used to evaluate GANs in more detail and thus guide multiobjective evolution more precisely, potentially resulting in more effective and creative solutions.

8 Conclusion

While GANs provide fascinating opportunities for generating realistic content, they are difficult to train and evaluate. This paper proposes an evolutionary metalearning technique, TaylorGAN, to optimize a crucial part of their design automatically. By evolving loss-functions customized to the task, dataset, and architecture, GANs can be more stable and generate qualitatively and quantitatively better results. TaylorGAN may therefore serve as a crucial stepping stone towards scaling up GANs to a wider variety and harder set of problems.

References

M. B. Alawieh, Y. Lin, Z. Zhang, M. Li, Q. Huang, and D. Z. Pan. GAN-SRAF: Sub-resolution assist feature generation using conditional generative adversarial networks. In *Proceedings of the 56th Annual Design Automation Conference (DAC)*, page 149. ACM, 2019.

M. Arjovsky, S. Chintala, and L. Bottou. Wasserstein generative adversarial networks. In D. Precup and Y. W. Teh, editors, *Proceedings of the 34th International Conference on Machine Learning (ICML)*, volume 70 of *Proceedings of Machine Learning Research*, pages 214–223, International Convention Centre, Sydney, Australia, 06–11 Aug 2017. PMLR. URL `http://proceedings.mlr.press/v70/arjovsky17a.html`.

A. Borji. Pros and cons of GAN evaluation measures. *Computer Vision and Image Understanding*, 179:41–65, 2019.

X. Chen, Y. Duan, R. Houthooft, J. Schulman, I. Sutskever, and P. Abbeel. InfoGAN: Interpretable representation learning by information maximizing generative adversarial nets. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29*, pages 2172–2180. Curran Associates, Inc., 2016.

W. Fedus, M. Rosca, B. Lakshminarayanan, A. M. Dai, S. Mohamed, and I. Goodfellow. Many paths to equilibrium: GANs do not need to decrease a divergence at every step. In *Proceedings of the Sixth International Conference on Learning Representations (ICLR)*, 2018. URL `https://openreview.net/forum?id=ByQpn1ZA-`.

S. Gonzalez and R. Miikkulainen. Improved training speed, accuracy, and data utilization through loss function optimization. In *Proceedings of the IEEE Congress on Evolutionary Computation (CEC)*, 2020.

S. Gonzalez and R. Miikkulainen. Optimizing loss functions through multivariate taylor polynomial parameterization. *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2021)*, 2021.

I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 2672–2680. Curran Associates, Inc., 2014. URL `http://papers.nips.cc/paper/5423-generative-adversarial-nets.pdf`.

J. J. Grefenstette and J. M. Fitzpatrick. Genetic search with approximate function evaluations. In *Proceedings of an International Conference on Genetic Algorithms and Their Applications*, pages 112–120, 1985.

I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville. Improved training of Wasserstein GANs. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5767–5777. Curran Associates, Inc., 2017. URL `http://papers.nips.cc/paper/7159-improved-training-of-wasserstein-gans.pdf`.

N. Hansen and S. Kern. Evaluating the CMA evolution strategy on multimodal test functions. In *International Conference on Parallel Problem Solving from Nature*, pages 282–291. Springer, 2004.

N. Hansen and A. Ostermeier. Adapting arbitrary normal mutation distributions in evolution strategies: The covariance matrix adaptation. In *Proceedings of IEEE international conference on evolutionary computation*, pages 312–317. IEEE, 1996.

N. Hansen and A. Ostermeier. Completely derandomized self-adaptation in evolution strategies. *Evolutionary computation*, 9(2):159–195, 2001.

J. Harer, O. Ozdemir, T. Lazovich, C. Reale, R. Russell, L. Kim, and p. chin. Learning to repair software vulnerabilities with generative adversarial networks. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems 31*, pages 7933–7943. Curran Associates, Inc., 2018.

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 I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, volume 30, pages 6626–6637. Curran Associates, Inc., 2017. URL `https://proceedings.neurips.cc/paper/2017/file/8a1d694707eb0fefe65871369074926d-Paper.pdf`.

G. E. Hinton and T. J. Sejnowski. Optimal perceptual inference. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 448–453. Citeseer, 1983.

P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros. Image-to-image translation with conditional adversarial networks. *arxiv*, 2016.

Y. Jin. Surrogate-assisted evolutionary computation: Recent advances and future challenges. *Swarm and Evolutionary Computation*, 1:61–70, 06 2011. doi: 10.1016/j.swevo.2011.05.001.

D. Kingma and M. Welling. Auto-encoding variational Bayes. In *Proceedings of the Second International Conference on Learning Representations (ICLR)*, 12 2014.

D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2015.

A. B. L. Larsen, S. K. Sønderby, H. Larochelle, and O. Winther. Autoencoding beyond pixels using a learned similarity metric. *arXiv preprint arXiv:1512.09300*, 2015.

M. Li, R. Xi, B. Chen, M. Hou, D. Liu, and L. Guo. Generate desired images from trained generative adversarial networks. In *Proceedings of the IEEE International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2019.

J. Liang, E. Meyerson, B. Hodjat, D. Fink, K. Mutch, and R. Miikkulainen. Evolutionary neural autoML for deep learning. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO)*, pages 401–409, 2019a.

J. Liang, E. Meyerson, B. Hodjat, D. Fink, K. Mutch, and R. Miikkulainen. Evolutionary neural automl for deep learning. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2019)*, 2019b. URL `http://nn.cs.utexas.edu/?liang:gecco19`.

X. Mao, Q. Li, H. Xie, R. Y. Lau, Z. Wang, and S. Paul Smolley. Least squares generative adversarial networks. In *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

X. Mao, Q. Li, H. Xie, R. Y. K. Lau, Z. Wang, and S. P. Smolley. On the effectiveness of least squares generative adversarial networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2018.

L. Metz, B. Poole, D. Pfau, and J. Sohl-Dickstein. Unrolled generative adversarial networks. *arXiv preprint arXiv:1611.02163*, 2016.

R. Miikkulainen, J. Liang, E. Meyerson, A. Rawal, D. Fink, O. Francon, B. Raju, H. Shahrzad, A. Navruzyan, N. Duffy, et al. Evolving deep neural networks. In *Artificial Intelligence in the Age of Neural Networks and Brain Computing*, pages 293–312. Elsevier, 2019.

M. Mirza and S. Osindero. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014.

T. Miyato, T. Kataoka, M. Koyama, and Y. Yoshida. Spectral normalization for generative adversarial networks. In *Proceedings of the Sixth International Conference on Learning Representations (ICLR)*, 2018.

J. Nash. Non-cooperative games. *Annals of Mathematics*, pages 286–295, 1951.

A. Odena, V. Dumoulin, and C. Olah. Deconvolution and checkerboard artifacts. *Distill*, 2016. doi: 10.23915/distill.00003. URL `http://distill.pub/2016/deconv-checkerboard`.

K. Pearson. On the criterion that a given system of deviations from the probable in the case of a correlated system of variables is such that it can be reasonably supposed to have arisen from random sampling. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 50(302):157–175, 1900.

A. Radford, L. Metz, and S. Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.

E. Real, A. Aggarwal, Y. Huang, and Q. V. Le. Regularized evolution for image classifier architecture search. In *Proceedings of the AAAI Conference on Artificial Intelligence*. AAAI, 2019.

S. Reed, Z. Akata, X. Yan, L. Logeswaran, B. Schiele, and H. Lee. Generative adversarial text to image synthesis. In M. F. Balcan and K. Q. Weinberger, editors, *Proceedings of the 33rd International Conference on Machine Learning (ICML)*, volume 48 of *Proceedings of Machine Learning Research*, pages 1060–1069, New York, New York, USA, 20–22 Jun 2016. PMLR. URL `http://proceedings.mlr.press/v48/reed16.html`.

A. R. Robertson. Historical development of cie recommended color difference equations. *Color Research & Application*, 15(3):167–170, 1990.

O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015. doi: 10.1007/s11263-015-0816-y.

R. Salakhutdinov and G. Hinton. Deep Boltzmann machines. In *Artificial Intelligence and Statistics*, pages 448–455, 2009.

H. Shahrzad, D. Fink, and R. Miikkulainen. Enhanced optimization with composite objectives and novelty selection. In *Artificial Life Conference Proceedings*, pages 616–622. MIT Press, 2018.

K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

P. Smolensky. Information processing in dynamical systems: Foundations of harmony theory. Technical report, Colorado University at Boulder Deptartment of Computer Science, 1986.

K. O. Stanley, J. Clune, J. Lehman, and R. Miikkulainen. Designing neural networks through evolutionary algorithms. *Nature Machine Intelligence*, 1:24–35, 2019. URL `http://nn.cs.utexas.edu/?stanley:naturemi19`.

B. Taylor. *Methodus incrementorum directa & inversa. Auctore Brook Taylor, LL. D. & Regiae Societatis Secretario.* typis Pearsonianis: prostant apud Gul. Innys ad Insignia Principis in . . . , 1715.

T. Tieleman and G. Hinton. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. *COURSERA: Neural networks for machine learning*, 4(2):26–31, 2012.

R. Tyleček and R. Šára. Spatial pattern templates for recognition of objects with regular structure. In *Proceeding of the German Conference on Pattern Recognition (GCPR)*, pages 364–374, Saarbrucken, Germany, 2013. Springer.

A. van den Oord, N. Kalchbrenner, L. Espeholt, K. Kavukcuoglu, O. Vinyals, and A. Graves. Conditional image generation with PixelCNN decoders. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29*, pages 4790–4798. Curran Associates, Inc., 2016. URL `http://papers.nips.cc/paper/6527-conditional-image-generation-with-pixelcnn-decoders.pdf`.

C. Villani. The Wasserstein distances. In *Optimal Transport*, pages 93–111. Springer, 2009.

V. Volz, J. Schrum, J. Liu, S. M. Lucas, A. Smith, and S. Risi. Evolving Mario levels in the latent space of a deep convolutional generative adversarial network. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO)*, pages 221–228. ACM, 2018.

T.-C. Wang, M.-Y. Liu, J.-Y. Zhu, A. Tao, J. Kautz, and B. Catanzaro. High-resolution image synthesis and semantic manipulation with conditional GANs. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8798–8807, 2018.

Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli. Image quality assessment: from error measurement to structural similarity. *IEEE Transactions on Image Processing*, 13(1), 2004.