Neuroevolution of a Multi-Generator GAN (Student Abstract)

Suraj Pandey
Indian Institute of Technology Guwahati, India
suraj18a@iitg.ac.in

Abstract

Evolutionary Algorithms (EA) have been leveraged to tackle the challenges faced while using GANs such as mode collapse, vanishing gradient, latent space search, etc. However, the existing techniques of using EA with GANs operate backpropagation and EA in isolation from each other, leaving ample room for further exploration. This paper creates a collaborative bridge between EA and GANs by exploring a neuroevolution method for utilising both EA and backpropagation-based optimisation, simultaneously, for a multi-generator GAN architecture. Experiments conducted using a standard dataset with variants of the proposed method highlight the towering impact of each of the components involved in the proposed method.

Introduction

Generative Adversarial Networks (GAN) (Goodfellow et al. 2014) have spanned across multiple domains including image translation, domain adaptation, style transfer, etc. While GANs produce remarkably good synthetic images, they suffer from multiple issues such as mode collapse and vanishing gradients (Wang et al. 2019). Further, since generators of GANs draw samples from a random distribution, GANs do not allow much control over the generated samples. Researchers have leveraged Evolutionary Algorithms (EA) to tackle these challenges. (Wang et al. 2019) evolved a population of generators, each having a different mutation objective, to compete with a discriminator. (Xue et al. 2022) followed a similar approach to evolve multiple generators. (Baeta et al. 2022) reformulated a GAN by replacing the traditional Deep Neural Network (DNN) based generator with a Genetic Programming (GP) based generator. EAs have also been utilised to search the latent space, that the generator uses, to control the generated samples (Roziere et al. 2021; Fernandes, Correia, and Machado 2020; Machín, Nesmachnow, and Toutouh 2021). While all these works utilise EA in their own ways for GANs, none of them attempts to use EA-based optimisation, in parallel with backpropagation, to tune the parameters of a GAN. Using EA and backpropagation together allows the GAN to conduct both exploration and exploitation of the search space, respectively. The reasons that make this task tough are enumerated as follows:

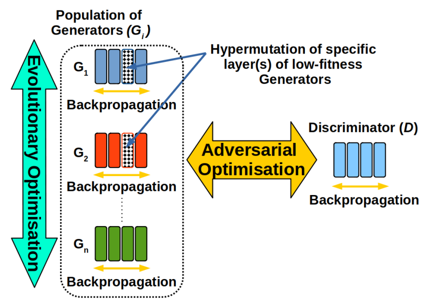

Figure 1: A schematic of the proposed multi-generator GAN architecture

1. Optimising the large number of parameters of a GAN using EA is a cumbersome task.
2. Using both EA and backpropagation on the same parameters of a DNN is an intricate task

This paper proposes a neuroevolution-based method that involves a population of generators evolved against a common discriminator. The subsequent sections describe the proposed method in detail followed by a brief analysis of the experimental setup and results.

Methodology

Fig. 1 depicts the proposed method, with a GAN comprising multiple generators, G_is, and a discriminator, D. Training this GAN involves two optimisation routines, *adversarial* and *evolutionary*, conducted across multiple generations. Each generation begins with the adversarial optimisation (O_A) of the parameters of all the G_is and the D. This process uses the real training dataset in a batch-wise manner. For each batch of real data, corresponding batches of fake data are generated by all the generators that draw input from a random distribution. Each G_i is optimised by utilising its corresponding fake data, the real data and the D. The D is optimised by utilising the fake data produced by all the G_is. This is followed by the evolutionary optimisation (O_E) routine that begins by calculating the fitness of each G_i. The fitness of the G_is is found as their corresponding generator loss values for a common random input. This is followed by a hypermutation process, wherein the the least half of the G_i population is chosen, based on their fitness values. For

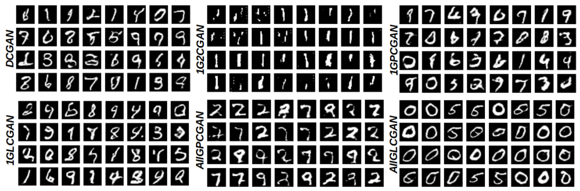

Figure 2: Images generated by variants of the proposed method and the DCGAN

each of the chosen G_is, an intermediate layer(s) is mutated by adding random Gaussian noise to its weights. The fitness of all the G_is are re-evaluated and the best G_i is chosen from the current generation. This completes both the adversarial and evolutionary optimisation routines for the current generation, yielding a next generation of G_is. The same process is followed for subsequent generations until the desired performance of the best generator is achieved. It is crucial to note that the O_A ensures the convergence of all the G_is to a common data distribution space, demarcated by the D. Whereas, the O_E ensures the divergence of G_is far from each other by utilising hypermutation, giving G_is a chance to search novel spaces in the optimisation landscape.

Experiments and Results

This section describes the experimental results obtained by using the proposed method and its variants on the MNIST (Lecun et al. 1998) dataset. The experimental setup was implemented using Tensorflow 2 and Keras frameworks using a *Cloud* GPU (NVIDIA T4(x2)). Experiments using five variants of the proposed method and a Deep Convolutional GAN (DCGAN) were conducted. The variants methods are described as follows:

1. ***1G2CGAN***: The (O_A) is conducted only for the best G_i whose weights are then copied to all G_is, except for the layer selected for mutation, before the (O_E) routine. The last two convolutional layers are selected for mutation.
2. ***1GPCGAN***: Same as $1G2CGAN$ with the penultimate convolution layer selected for mutation.
3. ***1GLCGAN***: Same as $1G2CGAN$ with the last convolution layer selected for mutation.
4. ***AllGPCGAN***: Follows the proposed method with the penultimate convolution layer selected for mutation.
5. ***AllGLCGAN***: Follows the proposed method with the last convolution layer selected for mutation.

Each method involved training across five generations having five epochs of (O_A) in each with a batch size of 32. For the sake of equivalence, DCGAN was trained for 25 epochs (without any generation parameter). Fig. 2 shows the images generated by using the best G_i of each method.

As seen in the figure, all of the proposed methods, except $1G2CGAN$, are able to generate images at par in quality with the DCGAN. It may be noted that the proposed methods are only allowed 5 epochs per generation for O_A based exploitation as opposed to 25 epochs allowed for DCGAN. Using only one layer (preferably the last) for mutation gives better results. Using multiple generators in both the O_A and O_E gives better quality images but tends to generate repetitive content (possibly mode collapse). Using only one layer for mutation stabilises the GAN convergence. The last two observations, along with an exploration of the images generated by the generator population of $AllGLCGAN$, are discussed in detail in the supplemental material of this paper.

Conclusion

This paper introduces a simple and efficient neuroevolution method for GAN, carried in conjunction with backpropagation. The paper analyses the impact of various strategies of neuroevolution, thereby serving as a preliminary study for clubbing O_A and O_E together. Owing to the complex task of merging O_A and O_E together, the experiments in this paper were restricted to MNIST dataset to serve as the preliminary findings. However, the proposed strategies shall be explored further with more complicated datasets.

References

Baeta, F.; Correia, J. a.; Martins, T.; and Machado, P. 2022. Exploring Expression-Based Generative Adversarial Networks. In *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, GECCO '22, 1862–1869. New York, NY, USA: Association for Computing Machinery. ISBN 9781450392686.

Fernandes, P.; Correia, J.; and Machado, P. 2020. Evolutionary Latent Space Exploration of Generative Adversarial Networks. In Castillo, P. A.; Jiménez Laredo, J. L.; and Fernández de Vega, F., eds., *Applications of Evolutionary Computation*, 595–609. Cham: Springer International Publishing. ISBN 978-3-030-43722-0.

Goodfellow, I.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A.; and Bengio, Y. 2014. Generative Adversarial Nets. In Ghahramani, Z.; Welling, M.; Cortes, C.; Lawrence, N.; and Weinberger, K., eds., *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc.

Lecun, Y.; Bottou, L.; Bengio, Y.; and Haffner, P. 1998. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11): 2278–2324.

Machín, B.; Nesmachnow, S.; and Toutouh, J. 2021. Evolutionary latent space search for driving human portrait generation. *2021 IEEE Latin American Conference on Computational Intelligence (LA-CCI)*, 1–6.

Roziere, B.; Teytaud, F.; Hosu, V.; Lin, H.; Rapin, J.; Zameshina, M.; and Teytaud, O. 2021. EvolGAN: Evolutionary Generative Adversarial Networks. In Ishikawa, H.; Liu, C.-L.; Pajdla, T.; and Shi, J., eds., *Computer Vision – ACCV 2020*, 679–694. Cham: Springer International Publishing. ISBN 978-3-030-69538-5.

Wang, C.; Xu, C.; Yao, X.; and Tao, D. 2019. Evolutionary Generative Adversarial Networks. *IEEE Transactions on Evolutionary Computation*, 23(6): 921–934.

Xue, Y.; Tong, W.; Neri, F.; and Zhang, Y. 2022. PEGANs: Phased Evolutionary Generative Adversarial Networks with Self-Attention Module. *Mathematics*, 10(15).