University of Caen Normandy
Department of Computer Science

Bibliographic Study of GANs: Advancements & Applications

Course: Advanced Deep Learning

Student Name: MESSILI Islem
Student Number: 22303045

Special Thanks to Professor: HÉRAULT Romain

January 23, 2025

Contents

1 Introduction

Generative Adversarial Networks (GANs), introduced by Ian Goodfellow et al. in 2014, pair a generator and a discriminator in a zero-sum game. The generator produces synthetic data, while the discriminator learns to distinguish real from fake. This adversarial setup has led to significant achievements in image and video synthesis, music composition, and text-to-image translation. Nevertheless, challenges like training instability and mode collapse persist. Recent variants (Wasserstein GAN [1], StyleGAN [2], BigGAN [3]) offer solutions that stabilize training and enhance output quality. Here, we survey these advancements and apply them to the JSB Chorales dataset to generate realistic and harmonically rich music.

2 Advancements in GANs

2.1 Improved Architectures

2.1.1 StyleGAN & StyleGAN2/3

NVIDIA's StyleGAN [2] introduced a way to disentangle high-level attributes (e.g., hair color, pose) from low-level details (e.g., texture). StyleGAN2 improved stability and removed artifacts, while StyleGAN3 focused on spatial coherence, making it suitable for high-quality videos and images. These improvements established StyleGAN as a key architecture for advanced image synthesis.

2.1.2 BigGAN

BigGAN [3] emphasizes large-scale image generation with high fidelity, employing techniques like group normalization and class-conditional layers. It achieves state-of-the-art results on benchmarks (e.g., ImageNet) and handles diverse categories effectively.

2.1.3 SAGAN (Self-Attention GAN)

Self-Attention GAN [4] uses attention mechanisms to model long-range dependencies in images, enabling the network to synthesize larger, more detailed images with improved coherence.

2.2 Training Stability Improvements

2.2.1 Wasserstein GAN (WGAN)

WGAN [1] replaces the original GAN loss with the Wasserstein distance, mitigating mode collapse and improving training reliability. This distance, also called the Earth Mover's Distance, helps the model converge more stably.

2.2.2 Spectral Normalization

Spectral Normalization [5] constrains the Lipschitz constant of the discriminator by normalizing its weights. This promotes stable convergence and better performance.

2.2.3 Gradient Penalty

WGAN-GP [6] enforces the Lipschitz constraint by penalizing gradients that deviate from a unit norm, thereby further stabilizing WGAN training.

2.3 Conditional GANs

2.3.1 cGANs

Conditional GANs [7] incorporate labels or other conditions (like text or images) into GANs, allowing targeted control over the generated outputs (e.g., image-to-image translation or text-to-image generation).

2.3.2 CycleGAN

CycleGAN [8] uses a cycle-consistency loss to learn mappings between two unpaired domains. It is especially useful for style transfer tasks such as transforming images into paintings.

2.4 GAN Variants for Specialized Tasks

2.4.1 VideoGAN

Video-oriented GANs (e.g., MoCoGAN [9]) maintain temporal consistency across frames, enabling coherent video synthesis.

2.4.2 MusicGAN

MuseGAN [10] adapts GANs for music generation using recurrent layers and attention to ensure both temporal coherence and musicality.

2.4.3 3D GANs

3D Generative Adversarial Models (e.g., [11]) generate shapes in voxel grids or point clouds, aiding in 3D modeling and reconstruction tasks.

2.5 Integration with Other Techniques

2.5.1 GANs + Transformers

Recent studies [12] incorporate Transformer architectures to address long-range dependencies in high-resolution image synthesis.

2.5.2 GANs + Diffusion Models

Diffusion approaches [13] offer stability advantages and broad coverage of the data distribution. They can complement or substitute GAN frameworks in some settings.

2.5.3 GANs + Reinforcement Learning

Combining GANs with RL [14] benefits applications like procedural content generation in games, enabling adaptive modeling of complex environments.

3 Dataset Description: JSB Chorales

3.1 Definition

The JSB (Johann Sebastian Bach) Chorales dataset is commonly used for music modeling and sequence learning. It comprises 382 harmonized chorales structured as chord sequences, where each time-step corresponds to a chord.

1. **Structure of the Dataset**
   - *Songs as Sequences:* Each piece is represented as a series of chords (e.g., (60, 72, 79, 88)), sometimes padded with zeros for uniformity.
   - *Variable Chord Lengths:* Chords may contain different numbers of notes (e.g., triads vs. four-note chords).
   - *Features:*
     - MIDI Note Numbers: Integer range from 21 to 108.
     - Velocity/Intensity: Normalized to [0, 1].
     - Timing: Normalized chord durations.
2. **Preprocessing for GAN**
   Chord data are normalized to [0, 1]:
   - *Pitch Normalization:*

$$\text{Normalized Pitch} = \frac{\text{MIDI Note} - \text{Min MIDI}}{\text{Max MIDI} - \text{Min MIDI}}$$

   - *Velocity and Timing Normalization:* Similar scaling to [0, 1].
3. **Dataset Statistics**
   - *Number of Songs:* 382 chorales, split into training, validation, and test sets.
   - *Sequence Lengths:* Vary from around 60 to over 100 chords.
   - *Chord Distribution:* Covers a wide range of intervals; up to four notes per chord.

3.2 Summary of the JSB Chorales Dataset

- **Core Idea:** 382 Bach chorales, each a chord-based sequence.
- **Structure:** Notes are grouped into chords, padded as needed; variable song lengths.
- **Features and Preprocessing:** MIDI pitch, velocity, and timing all normalized to [0, 1].
- **Use Cases:** Ideal for testing sequence modeling in music, including GAN-based approaches.

References

[1] Arjovsky, M., Chintala, S., & Bottou, L. (2017). *Wasserstein GAN.*

[2] Karras, T., Laine, S., & Aila, T. (2019). *A Style-Based Generator Architecture for Generative Adversarial Networks.*

[3] Brock, A., Donahue, J., & Simonyan, K. (2019). *Large Scale GAN Training for High Fidelity Natural Image Synthesis.*

[4] Zhang, H., Goodfellow, I., Metaxas, D., & Odena, A. (2019). *Self-Attention Generative Adversarial Networks.*

[5] Miyato, T., Kataoka, T., Koyama, M., & Yoshida, Y. (2018). *Spectral Normalization for Generative Adversarial Networks.*

[6] Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., & Courville, A. (2017). *Improved Training of Wasserstein GANs.*

[7] Mirza, M., & Osindero, S. (2014). *Conditional Generative Adversarial Nets.*

[8] Zhu, J. Y., Park, T., Isola, P., & Efros, A. A. (2017). *Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks.*

[9] Tulyakov, S., Liu, M., Yang, X., & Kautz, J. (2018). *MoCoGAN: Decomposing Motion and Content for Video Generation.*

[10] Dong, H. W., Hsiao, W. Y., Yang, L. C., & Yang, Y. H. (2018). *MuseGAN: Multi-track Sequential Generative Adversarial Networks for Symbolic Music Generation and Accompaniment.*

[11] Wu, J., Zhang, C., Xue, T., Freeman, W. T., & Tenenbaum, J. B. (2016). *Learning a Probabilistic Latent Space of Object Shapes via 3D Generative-Adversarial Modeling.*

[12] Esser, P., Rombach, R., & Ommer, B. (2021). *Taming Transformers for High-Resolution Image Synthesis.*

[13] Ho, J., Jain, A., & Abbeel, P. (2020). *Denoising Diffusion Probabilistic Models.*

[14] Lantao Yu, Weinan Zhang, Jun Wang, Yong Yu (2017). *SeqGAN: Sequence Generative Adversarial Nets with Policy Gradient.*