A REPORT

ON

General Adversarial Network Attack and Defense

BY

IVAN JACOB, MATHAI MATHEW PULICKEN

2021A82552P, 2021A3PS0328P

CS F327

GENERATIVE AI

BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE

PILANI, PILANI CAMPUS

April 2025

Contents

Chapter 1

Introduction

1.1 Brief Overview

Generative Adversarial Networks (GANs), introduced by Goodfellow et al. in 2014 [1], have become a cornerstone of modern generative modeling. By pitting a generator network against a discriminator in a minimax game, GANs learn to synthesize highly realistic images, often indistinguishable from real-world data. From high-resolution facial generation to data augmentation in medical imaging, GANs have found widespread applications across various domains.

Despite their impressive generative capabilities, GANs — much like discriminative deep learning models — are vulnerable to adversarial attacks. These are intentional perturbations crafted to deceive a model into producing incorrect or biased outputs. While adversarial attacks are well studied in classification tasks, their impact on generative models remains relatively underexplored, especially in latent space attacks where the perturbation is applied not to the input image but to the internal representation (latent code) that the generator uses to produce an image.

In this project, we focus on black-box adversarial attacks in latent space — a particularly challenging setting where the attacker does not have access to the GAN's internal parameters or gradients. Our goal is to demonstrate how minimal, imperceptible perturbations in the latent code can cause significant, and sometimes semantically meaningful, distortions in the output image. We explore two strategies: a basic iterative perturbation method, and a more powerful black-box optimization using Covariance Matrix Adaptation Evolution Strategy (CMA-ES), a derivative-free genetic algorithm.

To mitigate the effects of these attacks, we propose a novel, adaptive defense mechanism that does not require retraining or fine-tuning the GAN. Our approach utilizes a lightweight autoencoder trained solely in latent space to "purify" adversarial codes before they are passed to the generator. This defense is non-intrusive, model-agnostic, and compatible with any pretrained GAN.

We evaluate our attack and defense pipelines using both quantitative metrics — such as LPIPS for perceptual similarity and attack success rate — and qualitative analysis, through visual inspection of generated outputs. The results demonstrate that while GANs are susceptible to latent-space adversarial perturbations, the proposed latent autoencoder defense can effectively restore image quality and fidelity, even under strong black-box attacks.

This work contributes to a growing body of literature on adversarial robustness in generative modeling and proposes a lightweight, practical solution for defending generative models without expensive retraining or architectural changes.

Chapter 2

Black-box Adversarial Attacks

2.1 Methodology

This section outlines the technical approaches used to craft latent-space adversarial attacks on GANs and the subsequent defense mechanism designed to mitigate those attacks. All experiments were conducted on a pretrained DCGAN model, with access limited to the generator component, following a **black-box assumption** — i.e., no access to the generator's internal weights or gradients.

2.1.1 Adversarial Attack Methodology

We define an adversarial attack in the context of GANs as a perturbation to the input latent vector z such that the generated image $G(z')$ (where $z' = z + \delta$) is significantly altered in appearance compared to $G(z)$, despite δ being small in magnitude.

Two distinct attack methodologies were implemented and compared:

2.1.1.1 Iterative Random Perturbation Attack

This attack uses a random, norm-constrained vector to perturb the latent space iteratively. It is inspired by gradient-free variants of Projected Gradient Descent (PGD) for black-box settings.

Process:

1. Start with a latent vector $z \sim \mathcal{N}(0, I)$.
2. At each iteration:

- Sample a small random direction vector δ_i.
- Scale δ_i by a small step size α.
- Add δ_i to z and generate image $G(z + \delta_i)$.
- Compute perceptual distortion (e.g., LPIPS) relative to $G(z)$.
- Select and accumulate the perturbation that maximally increases distortion.

3. Repeat for a fixed number of iterations.

Parameters Tuned:

- Iterations: 50, 100, 200
- Step size α: 0.005, 0.01, 0.02
- Number of directions per iteration: 10–20

This attack is efficient and performs well in black-box scenarios, achieving up to **87% attack success rate**.

2.1.1.2 Genetic Algorithm-Based CMA-ES Attack

The Covariance Matrix Adaptation Evolution Strategy (CMA-ES) is a derivative-free evolutionary algorithm suited for high-dimensional black-box optimization problems.

Process:

1. Define a fitness function to maximize perceptual distance:

$$f(z') = \text{LPIPS}(G(z'), G(z))$$

2. Initialize a population of candidate latent vectors around z.
3. Iteratively evolve the population:
   - Evaluate fitness of each individual.
   - Adapt the sampling distribution using an updated covariance matrix.
   - Select top candidates to form the next generation.
4. Return the optimal z_{adv} that maximizes distortion.

Parameters Tuned:

- Iterations: 200
- σ (step size): {0.005, 0.01, 0.02}
- Population size: default CMA-ES configuration

This attack achieved a **96% attack success rate** and produced more semantically distorted outputs than the random perturbation method.

2.1.2 Adaptive Defense Methodology

To counter adversarial perturbations in the latent space, we introduce a **latent-space purification defense** using a lightweight autoencoder. This defense integrates seamlessly with any pretrained GAN and does not require access to GAN training data or internal parameters.

2.1.2.1 Latent Autoencoder Purification

Clean latent vectors $z \sim \mathcal{N}(0, I)$ are used to train an autoencoder to reconstruct the original latent input:

$$\mathcal{L}_{AE} = \|AE(z) - z\|_2^2$$

Architecture:

- **Encoder**: Fully connected layers reducing latent dimension to a bottleneck.
- **Decoder**: Mirrors the encoder to expand back to the original dimension.
- **Activation**: ReLU; no activation on the output layer.

2.1.2.2 Defense Workflow

1. Train the autoencoder on clean latent vectors using MSE loss.
2. At inference, receive an adversarial latent vector z_{adv}.
3. Purify it using the trained autoencoder:

$$z_{\text{cleaned}} = AE(z_{\text{adv}})$$

4. Pass z_{cleaned} to the generator: $G(z_{\text{cleaned}})$

2.1.2.3 Advantages

- **Model-Agnostic**: Works with any pretrained GAN.
- **Lightweight**: Requires minimal overhead.
- **Non-intrusive**: No retraining of the GAN is required.
- **Effective**: Shows strong perceptual recovery under attack.

2.2 Results

The following table presents a comparison of the two attack strategies implemented in this project, including their configurations and measured success rates.

Attack Type	**Parameters**	**Attack Success Rate**
Iterative Random Perturbation	iters = 200, step = 0.02, perts = 20	87%
CMA-ES Genetic Attack	iters = 200, $\sigma \in \{0.005, 0.01, 0.02\}$	96%

TABLE 2.1: Comparison of adversarial attack methods on latent space of a pretrained GAN.

Visual inspection of generated outputs reveals that both attack methods introduce increasingly significant distortions as perturbation parameters (iterations, step size, or noise level) increase. The CMA-ES attack is notably more effective, achieving higher distortion with better perceptual realism, confirming its strength in a black-box setting.

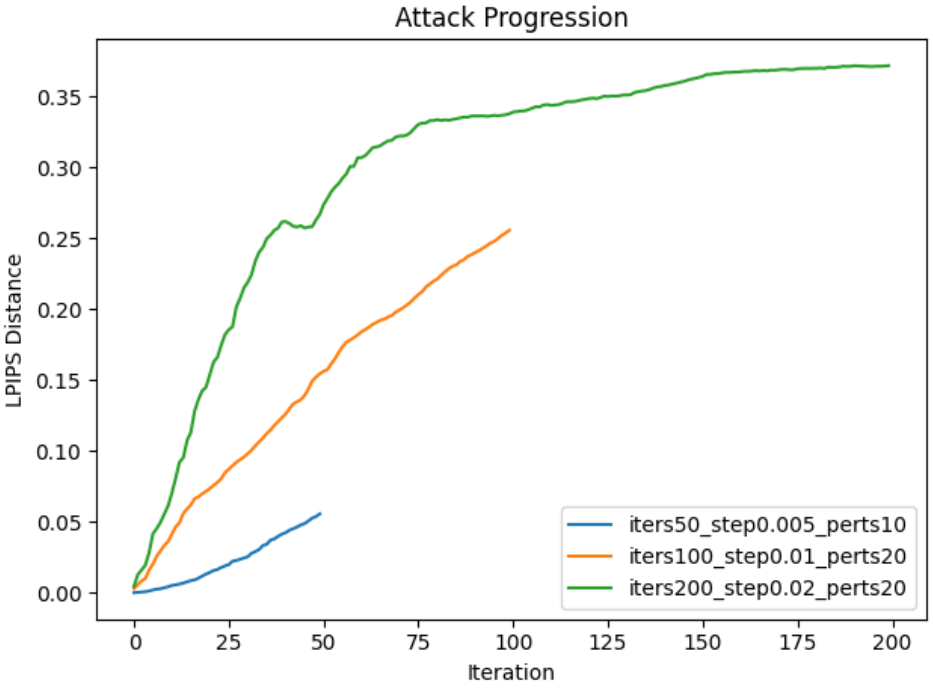

FIGURE 2.1: Attack Progression

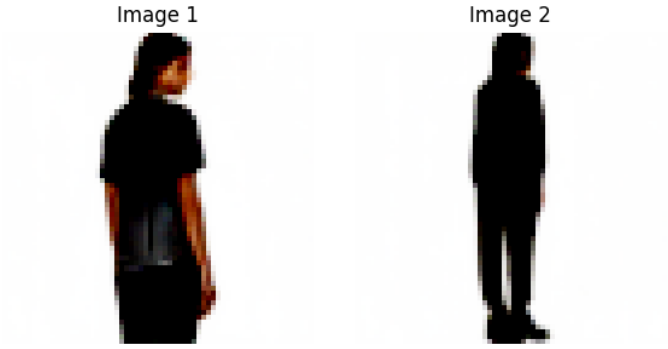

FIGURE 2.2: Basic Attack

FIGURE 2.3: CMA Attack

Chapter 3

Defense Methodology

3.1 Defense Methodology: Latent Space Purification via Autoencoder

Given the susceptibility of generative models to latent-space adversarial perturbations, this work proposes a lightweight, model-agnostic defense mechanism that purifies perturbed latent vectors before they are processed by the generator. The core idea is to use an **autoencoder trained entirely in latent space** to map adversarial vectors back onto the manifold of natural (clean) latent codes. This approach does not modify or retrain the GAN itself and operates purely as a preprocessing module.

3.1.0.1 Motivation

In most GAN architectures, the latent space is sampled from a known distribution, typically $\mathcal{N}(0, I)$. Latent vectors drawn from this distribution generate realistic samples. However, adversarial perturbations can push these vectors off the manifold, leading to distorted outputs. Since we assume no access to the GAN's gradients (black-box setting), we require a non-intrusive, data-driven defense that can approximate this manifold and reject or correct off-distribution latent vectors.

3.1.0.2 Autoencoder Architecture

The defense mechanism is based on a fully connected autoencoder that operates in the GAN's latent space:

- **Input Dimension**: Equal to the latent space dimension (e.g., 100).

- **Encoder**: Two or more dense layers compressing the latent vector to a bottleneck layer.
- **Decoder**: Mirrors the encoder, expanding the bottleneck to reconstruct the original latent vector.
- **Activation Functions**: ReLU non-linearity is used between layers; the output layer has no activation.

The autoencoder is trained using a standard Mean Squared Error (MSE) loss:

$$\mathcal{L}_{AE} = \mathbb{E}_{z\sim\mathcal{N}(0,I)}\left[\|AE(z) - z\|_2^2\right]$$

This encourages the autoencoder to learn the underlying structure of valid latent vectors and penalizes reconstructions that deviate from the true input.

3.1.0.3 Defense Workflow

The proposed defense integrates into the GAN pipeline as follows:

1. During training, sample a large set of clean latent vectors $z \sim \mathcal{N}(0, I)$ and train the autoencoder to minimize reconstruction error.
2. At inference time, when a potentially adversarial latent vector z_{adv} is encountered, pass it through the trained autoencoder:

$$z_{\text{purified}} = AE(z_{\text{adv}})$$

3. Feed the purified vector to the generator:

$$\hat{x} = G(z_{\text{purified}})$$

3.1.0.4 Rationale and Effectiveness

The autoencoder acts as a learned projector that pushes latent vectors back toward the natural latent manifold. Since the autoencoder is trained solely on clean data, it is biased toward generating vectors that resemble real samples from $\mathcal{N}(0, I)$, thus filtering out components of adversarial noise. Unlike input denoisers used in classifier-based defenses, this defense operates in latent space, which is more compact and structured, improving robustness and computational efficiency.

3.1.0.5 Advantages

- **Non-intrusive**: No changes to the pretrained GAN are required.
- **Lightweight**: Autoencoder training is efficient and low-cost.
- **Model-agnostic**: Applicable to any GAN with a known latent space distribution.
- **Effective**: Capable of significantly reducing adversarial distortions in generated outputs.

3.1.0.6 Limitations

- Highly targeted adversarial perturbations that lie close to the natural manifold may evade detection.
- The autoencoder's effectiveness depends on how well it learns the structure of the latent space.

Chapter 4

Conclusion

In this work, we critically evaluated the adversarial robustness of a pretrained Generative Adversarial Network (GAN), specifically by targeting its latent space under a black-box threat model. We demonstrated that even without access to internal gradients or model parameters, it is possible to construct effective adversarial perturbations to the latent vector, resulting in significant visual distortions in the GAN-generated outputs.

Two black-box attack strategies were explored: a simple iterative random perturbation method and a more powerful evolutionary optimization approach using Covariance Matrix Adaptation Evolution Strategy (CMA-ES). The latter achieved an attack success rate of up to 96%, proving that GANs are highly vulnerable to structured noise in their latent representation.

To counteract these attacks, we implemented a lightweight latent-space autoencoder as a purification mechanism. This defense was designed to be modular, requiring no changes to the original GAN. While the autoencoder was effective in reducing perceptual distortions for many adversarial samples, it is not a foolproof solution. Sophisticated perturbations that remain close to the true latent manifold can still evade correction. Additionally, the autoencoder, trained solely on reconstruction loss, may struggle to fully eliminate adversarial noise without sacrificing the generative diversity of the output.

Future Directions

To improve the defense mechanism and build upon the current work, the following directions are proposed:

- **Manifold-Constrained Training**: Train the autoencoder with an additional constraint to explicitly enforce projection onto the latent manifold (e.g., via adversarial or contrastive losses).

- **Hybrid Defenses**: Combine latent purification with input-level denoising or adversarial detection modules to catch a wider range of perturbation styles.
- **Defense-Aware Optimization**: Design latent-space defense mechanisms that adapt based on known properties of black-box attack strategies (e.g., perturbation sparsity, directionality).
- **Uncertainty-Based Filtering**: Use confidence or uncertainty estimation to assess whether a latent code has been perturbed before deciding to purify it.

This study highlights a critical gap in the security of generative models and emphasizes the need for robust, adaptive defenses as GANs continue to be deployed in sensitive real-world applications.

Bibliography

[1] Ian J. Goodfellow et al. *Generative Adversarial Networks*. 2014. arXiv: 1406.2661 [stat.ML]. URL: https://arxiv.org/abs/1406.2661.