Comparative Analysis of Conditional and Super-Resolution Generative Adversarial Networks (CGANs vs SRGANs) for Enhancing Image Resolution

Anh Pham

***Abstract*— This paper investigates the use of Generative Adversarial Networks (GANs) for image super-resolution, focusing on a comparative analysis of Conditional GANs and Super-Resolution GANs. Specifically, this study examines the efficiency of these models in terms of inference speed when enhancing low-resolution images to high-resolution outputs. By conditioning the GAN on low-resolution inputs, CGANs are adapted for super-resolution, allowing a side-by-side comparison with SRGANs, which are specifically designed for this task. This work aims to evaluate the computational trade-offs between these models to inform their application in real-time settings.**

***Index Terms*— image super-resolution; interference time; generator; discriminator**

I. Topic Introduction

Generative Adversarial Networks (GANs) are a type of deep learning architecture used to train a generative model, where the objective is to generate new data that resembles the training data. GANs generate new, realistic data samples that resemble a given dataset, making them powerful for image synthesis, image-to-image translation, and super-resolution tasks in computer vision. These applications make GANs valuable tools in fields like entertainment, healthcare, autonomous driving, and more.

A GAN architecture model involves two sub-models: a generator model for generating new examples and a discriminator model for classifying whether generated examples are real or fake.

- Generator model: This model generates new data samples by learning to map random noise to the data distribution of the target dataset. In image generation, it takes random noise (often a vector of random values) and transforms it into an image.
- Discriminator model: This model acts as a classifier, distinguishing between real images from the training dataset and fake images created by the generator.

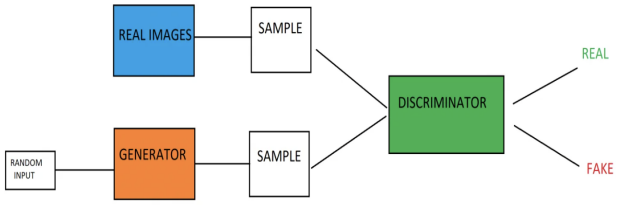
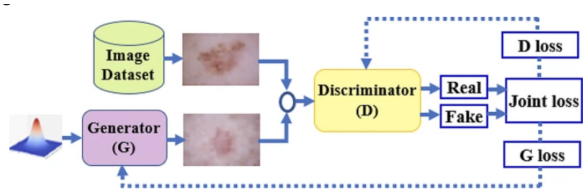


Fig. 1. A high-level diagram of the GAN Generator-Discriminator Infrastructure



A typical GAN architecture

Fig. 2. A typical GAN Generator-Discriminator Infrastructure with its loss functions [1]

In a Generative Adversarial Network (GAN), the underlying mechanism involves the generator striving to produce data that can deceive the discriminator, while the discriminator attempts to differentiate between real and generated data. This adversarial training process fosters continuous improvement in both the generator and discriminator, making them two competitive networks that force each other to be better. This mechanism is reinforced upon examining the GAN loss function:

$$\min_G \max_D V(D,G) = \mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z})))].$$

Fig. 3. Standard GAN loss function (min-max GAN loss)

The foundation of GAN is a min-max game in which:

- **Generator**: Minimizes $\log(1 - D(G(z)))$ or equivalently maximizes $-\log D(G(z))$, aiming to fool the discriminator into classifying generated data as real.
- **Discriminator**: Maximizes $\log D(x)$ for real data and $\log(1 - D(G(z)))$ for generated data to distinguish real from fake.

This adversarial interaction drives iterative improvement.

II. Existing Methods

GANs have been widely adopted for image super-resolution due to their ability to generate high-quality outputs. Conditional GANs (CGANs) extend traditional GANs by incorporating auxiliary conditions, such as image-specific features or labels, to guide the generation process. These conditions help CGANs achieve more tailored and accurate outputs. On the other hand, Super Resolution GANs (SRGANs) introduce perceptual loss functions, combining pixel-wise and feature-space losses, to produce visually realistic super-resolved images. While effective, SRGAN's training process remains challenging due to potential instability between

the generator and discriminator networks. Both SRGAN and CGAN provide distinct advantages for super-resolution tasks, but their inference speed and performance have not been directly compared in practical applications. This paper investigates these models to evaluate their efficiency and suitability for real-world super-resolution applications.

A. Conditional GANs

Conditional GANs (CGANs) are an extension of this standard GAN model that incorporate additional information, or conditions, into both the generator and discriminator networks. They can be used for a wide range of purposes such as conditioning on labels to generate specific classes of images (e.g. dogs, cats).

In a CGAN, the generator takes both random noise and the condition as inputs. This condition can be any extra information that provides specifications as to what the generated output should look like, such as a class label, a feature vector, or another low-resolution version of an image. The discriminator also receives both the generated (or real) data and the condition, allowing it to evaluate whether the generated data aligns with the given condition. This setup helps the discriminator learn to recognize whether the generator's output is both realistic and correctly conditioned.

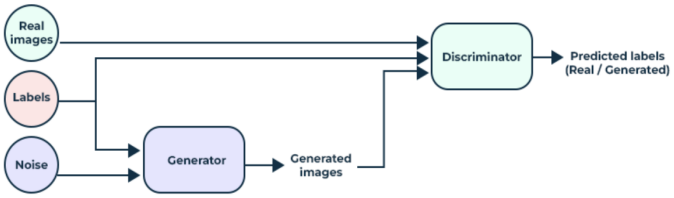


Fig. 4. CGAN Generator-Discriminator Infrastructure

The loss function for a Conditional GAN extends the standard GAN. It fundamentally plays the same min-max game but using conditional probability for both the generator and the discriminator. [2] [3]

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

B. CGANs for Image Super-Resolution

While CGANs are typically not used for image super-resolution, Qiao et al. (2019) suggests a CGAN model that can be used in a highly efficient manner for super-resolution purposes. This research presents using a type of CGAN specifically for super-resolution called SRCGAN or Super-Resolution Conditional GAN. By conditioning the discriminator on ground-truth images, SRCGAN inputes a low-resolution image to the generator and upscales it to a high-resolution version. The generator uses a deep residual network with convolutional layers and learns to upscale low-resolution (LR) images to high-resolution (HR) outputs while the discriminator distinguishes real from generated images. This mechanism provides the generator with feedback to improve image resolution quality. [4]

C. Super-Resolution GANs

Super-Resolution GANs (SRGANs) are a subtype of Generative Adversarial Networks that are specifically built for image super-resolution ie. increasing the resolution of a low-resolution image to create a higher-quality, more detailed version. They achieve this goal by having the generator model generate finer details and textures of the inputted image. [5]

SRGANs use a GAN framework tailored to generate realistic high-resolution images from low-resolution inputs, specifically trained on image quality. The generator takes in a low-resolution image as input and attempts to generate a high-resolution output by learning patterns and fine details from the data. It uses techniques like residual blocks and upsampling layers to enhance the resolution and add realistic texture to the output. The discriminator evaluates the generated high-resolution image and tries to distinguish it from real high-resolution images. It provides feedback to the generator about how realistic its output is, encouraging the generator to produce more convincing images.

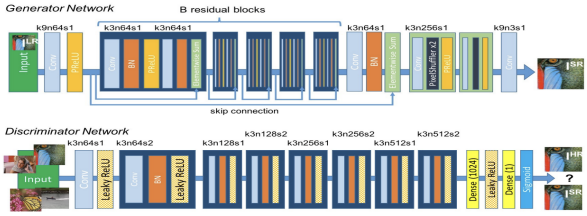


Fig. 5. GAN Generator-Discriminator Infrastructure [5]

The SRGAN loss function distinguishes itself from standard GAN and CGAN by incorporating content loss, derived from feature comparisons in a pre-trained VGG (Visual Geometry Group) network, alongside the traditional adversarial loss. This perceptual loss ensures that the generated image not only fools the discriminator (adversarial goal) but also closely matches the high-resolution image in terms of perceptually relevant features, rather than just pixel-level similarity. This combination enables SRGAN to achieve visually superior super-resolution results. The description and mathematical principles of SRGAN is beyond the scope and the intended length of this short research paper, thus more details can be found in the source cited. [6]

III. Current Limitations

A. Existing Studies on GAN Performance Comparisons

Based on a review of existing literature via Google Scholar, it appears that the body of research focused on performance comparisons among GANs remains relatively limited. One such study by B. S. et al. (2023) found that the performance of various GAN architectures — such as DCGAN, SRGAN, and CGAN — varies significantly depending on the dataset and architecture used, with no single model consistently outperforming others across all tasks [7]. In Jozdani et al. (2022), SRGAN and CGAN

are compared for image generation in remote sensing. This comparison suggested that SRGANs excels in tasks like super-resolution, while CGAN is more suitable for scenarios requiring controlled image outputs through conditional inputs [8]. The results of previous research are summarized in the table below.

TABLE I
SUMMARY OF COMPARATIVE STUDIES ON GANS

Study	Findings
B. S. et al. (2023) *A Comparative Analysis on the Effectiveness of GAN Performance* [7]	Evaluated GANs such as DCGAN, SRGAN, and CGAN. Found performance depends on dataset and architecture; no single model excels across all tasks.
Jozdani et al. (2022) *A Review and Meta-Analysis of Generative Adversarial Networks and Their Applications in Remote Sensing* [8]	Compared SRGAN and CGAN for remote sensing. SRGAN excels in super-resolution, while CGAN is better for controlled image generation using conditional inputs.

In the subfield of Generative Adversarial Networks (GANs) for image generation, several key limitations persist, particularly in balancing image quality and computational efficiency. These trade-offs are important for future research including mine. Models like SRGAN excel in high-resolution image generation but are computationally demanding, making them impractical for real-time applications. CGANs provide greater control through conditioning on additional inputs but often struggle with image quality when the conditional information is sparse or poorly defined.

Another significant challenge lies in generalization across different datasets. GANs, including SRGAN and CGAN, are highly dependent on the dataset they are trained on. Thus performance can degrade when applied to new, unseen data or in domains where labeled data is limited.

IV. RESEARCH IDEAS

The present paper evaluates the computational trade-offs between CGANs and SRGANs for enhancing image resolution. Using the MNIST dataset, the study compares the efficiency and image quality of both models, with SRGAN modified to handle low-to-high-resolution transformations. The goal is to assess the balance between computational efficiency and image quality. By investigating these models on a well-established dataset like MNIST, the research provides insights that can inform the selection of GAN architectures in real-world applications under resource constraints.

V. RESEARCH METHODS

To achieve the goal, I will leverage TensorFlow and NumPy for easy loading and manipulation of the MNIST dataset, while implementing both CGAN and SRGAN models based on existing GitHub code. Techniques such as perceptual loss for SRGANs will be incorporated to assess how well these models balance high-resolution output with computational load.

For comparisons to be made, the objects of such comparison need to be comparable. The proposed method involves modifying CGANs into SRGANs to compare their capabilities in generating high-resolution images from low-resolution inputs. This approach is sensible because both models share a similar underlying architecture, consisting of a generator and a discriminator.

- **Architecture**: Both CGAN and SRGAN share the same generator-discriminator structure. While CGAN generates images from random noise and class labels, SRGAN generates high-resolution images from low-resolution inputs.
- **Scale**: Both models are applied to the MNIST dataset images (28x28 pixels), allowing a fair comparison of computational efficiency, network complexity, and training time.
- **Discriminator**: Both models use a similar discriminator to distinguish between real and generated images, evaluating either digit outputs (CGAN) or enhanced high-resolution images (SRGAN), enabling consistent comparison.

VI. CONCEPT TO CODE

GitHub source code folder link:
GAN.ipynb [9]

A. Dataset Description

This study uses the Modified National Institute of Standards and Technology (MNIST) dataset. It is a widely recognized benchmark for machine learning and computer vision tasks. The details are as follows:

- **Content:** 70,000 grayscale images of handwritten digits, ranging from 0 to 9.
- **Image Size:** Each image has dimensions of 28×28 pixels.
- **Purpose:** Primarily used for training machine learning models, particularly for digit classification tasks.
- **Data Split:** The dataset is divided into 60,000 images for training and 10,000 images for testing.
- **File Format:** Stored in the `.npz` format (NumPy compressed file), making it easily accessible in Python through libraries like TensorFlow or NumPy.

B. Run Instructions

Run the first code block for the CGANs implementation. Then run the second code block for the SRGANs implementation. Compare the time output and observe the quality of the image generated. Assess the trade-offs.

C. Connection to Proposed Research Idea

This code compares two GANs implementation - CGANs and SRGANs on the same dataset and demonstrated the time trade-offs when using the two models. The code has an implementation of SRGANs based on the upscaling mechanism from CGANs to SRGANs. The perceptual loss function implementation was unsuccessful due to mismatch between expected input and the actual output of the function. Further

understanding of mathematical and statistical principles are needed to correctly implement the loss function to replicate a comparable version of SRGANs from our original model CGANs.

D. Technical Implementations for the Next Step

To compare the performance of the CGAN and SRGAN models, particularly in terms of training time, a few strategies in the next step are recommended to measure and track performance efficiently. The code is partially implemented and cannot yet to be run until the SRGANs implementation is successful.

1) Time Tracking with Python's `time` Module
   - *Start and End Time:* Track the start and end times for each training loop to compare the total time taken for training each model.
   - *Epoch-wise Comparison:* Track the time per epoch for both CGAN and SRGAN to see how long each epoch takes for training.
   - *Functionality:* This allows us to isolate where the training time is spent (e.g., generator vs. discriminator updates, image generation, etc.).

VII. Results & Discussion

The code implementation, if successfully run in the future, will demonstrate that one of the two models performed better on the specific dataset MNIST. It is important to consider the nature and characteristic of the MNIST dataset and how they play into the experimental result. MNIST dataset is widely used for training machine learning models, particularly for digit classification, but it may not be the best choice for evaluating image resolution enhancement that may demand high-quality, complex image outputs. MNIST consists of 70,000 grayscale images of handwritten digits (28x28 pixels), which are relatively simple and low-resolution compared to the high-resolution image generation tasks often associated with real-time applications in fields like computer vision or remote sensing. However, even if MNIST was a relatively reasonable choice of dataset within our runtime and computational constraints, the stability of CGANs and SRGANs models have to be tested across diverse datasets for a direct conclusion from our demonstration to be reliable. Based on previous literature, it is plausible that our experiment would largely be inconclusive if not given all comprehensive factors such as the scale and nature of datasets and the specific infrastructure used to run GANs.

VIII. Additional Project Materials

- Presentation Deck: GANs - Google Slide Presentation

References

[1] E. Goceri, "Gan based augmentation using a hybrid loss function for dermoscopy images," *Artif Intell Rev 57*, vol. 234, 2024.

[2] H. Dwivedi, "Understanding gan loss functions," *neptune.ai*, 2023.

[3] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," *Advances in neural information processing systems*, vol. 27, 2014.

[4] J. Qiao, "Image super-resolution using conditional generative adversarial network," *IET Image Processing*, vol. 13, pp. 2673–2679, 2019.

[5] C. Ledig, "Photo-realistic single image super-resolution using a generative adversarial network," *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 105–114, 2017.

[6] D. Chakraborty, "Super-resolution generative adversarial networks (srgan)," 2022.

[7] C. BS, M. Shreyas, I. Karanth, B. S, A. R. KP, and G. S, "A comparative analysis on the effectiveness of gan performance," pp. 1–8, 2023.

[8] S. Jozdani, D. Chen, D. Pouliot, and B. Alan Johnson, "A review and meta-analysis of generative adversarial networks and their applications in remote sensing," *International Journal of Applied Earth Observation and Geoinformation*, vol. 108, 2022.

[9] GeeksforGeeks, "Conditional gans (cgans) for image generation," 2024. [Online]. Available: https://www.geeksforgeeks.org/conditional-gans-cgans-for-image-generation/