**MOSCOW STATE UNIVERSITY**

The Faculty of Computational Mathematics and Cybernetics

# Generative Adversarial Networks (GANs) for Image Synthesis and Enhancement

Yudina Diana

217 group

**Moscow**

**2025**

## Table of contents

## Introduction

Generative Adversarial Networks (GAN) are a large class of generative models whose common feature is that they are trained simultaneously with another network that tries to distinguish generated objects from real ones.These networks allow you not only to create realistic images from scratch, but also to improve the quality of existing images. We will look at the basic principles of GAN operation, their architecture, application for image synthesis and enhancement, as well as the prospects for the development of this technology. We will also give examples of several types of practical tasks in which generative-adversarial networks are used.

## Theoretical foundations of generative models

Generative models are a class of machine learning algorithms capable of generating new data similar to the initial distribution of the training dataset. Key approaches in this area are Autoencoders, variational autoencoders (VAE), and generative adversarial networks (GANS).

### The basic idea of GAN

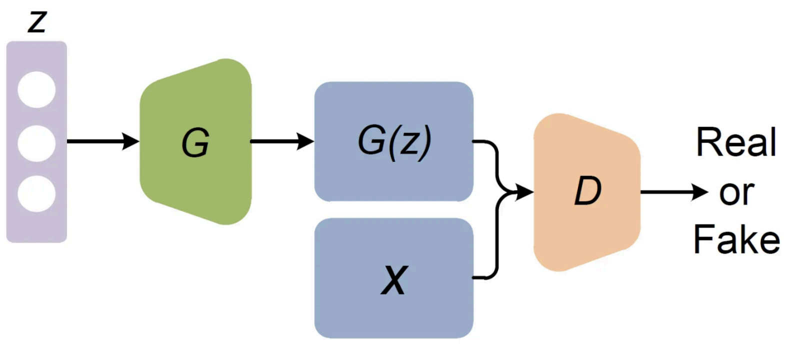
GANS were proposed by Ian Goodfellow in 2014. The basic concept is that two neural networks (a generator and a discriminator) are trained together, competing with each other. The generator creates fake images in an attempt to deceive the discriminator, while the discriminator tries to distinguish the real images from the generated ones.

## GAN Architecture

The GAN architecture consists of two neural networks:

*Generator:* Creates synthetic data from random noise to produce data so realistic that the discriminator will not be able to distinguish it from real data.

*Discriminator:* Acts as a critic, assessing whether the data obtained is real or fake. They use competitive learning to create artificial data that is identical to real data.



**Generator Model (G)**

The generator creates fake data (for example, images) using random noise. It is a neural network that takes a vector of random numbers (noise) as input and converts it into an image.

- At the beginning of training, the generator creates chaotic images that are far from reality.
- Gradually, he learns to create better and better images that can deceive the discriminator.
- Usually, the generator consists of convolutional and fully connected layers, and at the last stage, the Tanh activation function is applied.

*GANS are optimized using a minimax loss function:*

$$\text{Binary Cross-Entropy Loss} = -\frac{1}{N}\sum_{i=1}^{N} y_i\,(\log(p_i)) + (1-y_i)(\log(1-p_i))$$

**The discriminator model (D)**

A discriminator is an ordinary neural classifier network that receives real images (from a training sample) and generated images (from a generator) as input. His task is to learn to distinguish between them.

- At the beginning, the discriminator easily distinguishes fake images from real ones.
- As he learns, he encounters improved fakes and is forced to adapt.

- At the output, the discriminator outputs the probability that the input image is real.

## Hardware Requirements for GANs

Training GANs, especially for high-resolution image generation, requires substantial computational power. The following hardware components are typically used for training and deploying GAN models:

1. **Graphics Processing Units (GPUs)**:
   GPUs are essential for training deep learning models, including GANs. They significantly accelerate the training process by parallelizing computations. Modern GPUs like NVIDIA's **RTX 3090**, **Tesla V100**, or **A100** are commonly used for high-performance tasks. The use of **CUDA** (Compute Unified Device Architecture) allows deep learning frameworks (like TensorFlow, PyTorch, etc.) to perform efficient computations on GPUs.
2. **Tensor Processing Units (TPUs)**:
   TPUs, developed by Google, are specialized hardware designed to accelerate machine learning workloads, particularly deep learning models. TPUs can offer better performance than GPUs in some cases and are available via **Google Cloud** for large-scale computations.
3. **High-performance CPUs**:
   While GPUs are crucial for deep learning, **CPUs** still play a significant role in general-purpose computations and managing other tasks in the workflow. Multi-core processors from manufacturers like Intel and AMD are commonly used to handle these tasks.
4. **Large Memory (RAM)**:
   GANs, especially those used for high-resolution image generation, require large amounts of memory. Typically, 32GB or more of **RAM** is needed to train models on datasets with large image sizes or complex architectures.

5. **Storage**:

   **Solid-state drives (SSDs)** are preferred for storing large datasets and trained models due to their faster read/write speeds. Training a GAN can generate substantial data, so having sufficient storage space is necessary. Cloud storage solutions are often used for scalable storage needs.
6. **Networking**:

   Fast and reliable **networking** is essential for training GANs in distributed systems or cloud environments. Efficient data transfer speeds are critical for managing large datasets and sharing model weights between multiple nodes.
7. **Powerful Workstations or Servers**:

   Depending on the size of the model and dataset, powerful **workstations** or **dedicated servers** with multiple GPUs (often called **GPU farms**) are used to handle the computations involved in training GANs.

## Image synthesis

Image synthesis using generative adversarial networks (GANS) is the process of creating new images that look realistic but are not copies of existing data. GANS can generate people's faces, objects, scenes, paintings, and even stylized images based on text descriptions.

It involves creating images based on existing data or models.

*Examples of synthesis tasks:*

- Over-resolution. Zoom in low-resolution images to higher resolution ones.
- Painting over. Filling in missing or damaged parts of the image.
- Style transfer. Applying the style of one image to the content of another.

### Examples of using image synthesis with GAN

*Medicine*

- Generation of medical images (MRI, X-rays) to train doctors and improve diagnostics.

- Improve the quality of low-resolution medical images.

*Movies and games*

- Create unique 3D characters and textures for video games.
- Generate photorealistic environments that do not require long manual modeling.

*Art and Design*

- GAN can generate new paintings in the style of famous artists.
- Automatic stylization of images in various artistic genres.

*Fashion*

- Create new designs of clothes and accessories.
- Generation of virtual models for fitting clothes.

## The pros and cons of using a Generative Adversarial Network (GAN)

Generative adversarial networks (GANS) have become a popular tool for generating images, videos, and other data. However, they have their advantages and disadvantages, as well as alternative methods that sometimes work better.

**Advantages of GAN**

- High–quality generation - GAN creates photorealistic images that are difficult to distinguish from the real ones.
- No need for marked–up data - learning takes place without manual marking.
- Flexibility – GANS can change styles, finish and enhance images.
- Generating different types of data is used to create text, video, music, and speech.
- Image Enhancement and restoration – GAN improves the resolution and quality of photos.

**Disadvantages of GAN**

- Difficulty of learning – GAN is difficult to train, since the network consists of two parts (a generator and a discriminator) that compete with each other.
- Mode Collapse – the network can start creating the same type of images, losing diversity.
- High computing costs – require powerful video cards

**Alternatives to GAN**

- Variational Autoencoders (VAE)

  *Advantage:* they are easier to learn and allow you to manage the generation process.

  *Disadvantage:* they create less detailed and blurred images.
- Diffusion models (DALL·E 2, Stable Diffusion)

  *Advantage:* high quality, good handling.

  *Disadvantage:* generation takes longer and requires powerful equipment.
- Flow models (Flow-based models, for example, Glow)

  *Advantage:* good control over generation.

  *Disadvantage:* they do a worse job with photorealism than GAN.

## An example of website implementation using GAN technology

1. *The main components of the project The project will include several parts:*
   - Using GAN to generate images.
   - A website for user interaction interface.
   - GAN integration with the web server.
2. *GAN for image generation*

Generative Adversarial Networks (GAN) is a class of machine learning models consisting of two neural networks: A generator that generates data (in this case, images). Discriminator, which tries to distinguish real images from fake ones. To generate photos of people, you can use an already trained GAN model, for example:

- StyleGAN2 is one of the most popular models for generating photorealistic images of people. She uses the GAN architecture and creates high-quality images of people.
- BigGAN is a more powerful model for generating large and detailed images.

3. *Website creation*

To create a website, we will use html css javascript. In order for users to interact with the system, you will need to create a web interface. This website should have a button that will generate an image when clicked.

*Necessary tools:*

- HTML/CSS — to create a website interface.
- JavaScript (or a framework, for example, React) — for interacting with the user and sending requests to the server to generate images.
- The backend (server side) is Python Flask or Django, which will process requests from the user and generate images using the GAN model.

```
<button onclick="generateImage()">Generate an image</button>
<div id="image-container"></div>
<script>
 function generateImage() {
   fetch('/generate')
     .then(response => response.json())
     .then(data => {
             document.getElementById("image-container").innerHTML = `<img
src="${data.image_url}" />`;
     });
 }
```

A template for creating a button on a web page that will send a request to the server to generate a new image.

```
from flask import Flask, jsonify
import torch
from model import generator  # Assuming the GAN model is saved in the
'model.py' file
app = Flask(__name__)
```

```
@app.route('/generate', methods=['GET'])
def generate_image():
    # Generate an image using the GAN model
    with torch.no_grad():
        generated_image = generator(torch.randn(1, 100))  # Input for the generator could be a random vector
    # Save the generated image on the server
    image_path = 'static/generated_image.png'
    generated_image.save(image_path)
    # Return the URL of the generated image as a JSON response
    return jsonify(image_url=image_path)

if __name__ == "__main__":
    app.run(debug=True)
```

A small implementation of a Python server (Flask) that will accept a request from the user and generate an image by transmitting it in response.

Below are people whose faces are created by the GAN neural network, which in reality does not exist.

## Conclusion

Generative Adversarial Networks (GANs) have fundamentally transformed the landscape of artificial intelligence, especially in the domains of image synthesis, enhancement, and data generation. Their ability to generate highly realistic images from random noise or modify existing images to improve their quality has made them indispensable in fields ranging from art and entertainment to medical imaging and autonomous systems.

The architecture of GANs, comprising a generator and a discriminator, creates a competitive environment where both networks improve iteratively, leading to the generation of images that are indistinguishable from real ones. This makes GANs particularly powerful for tasks like photo-realistic image generation, data augmentation, and even creative tasks like art generation and style transfer.

However, GANs are not without challenges. Issues such as mode collapse, training instability, and the lack of reliable evaluation metrics can complicate the development and deployment of GAN-based systems. The generator might sometimes produce a limited set of outputs, failing to fully explore the diversity of possible images. Additionally, training GANs requires significant computational resources and expertise in balancing the generator and discriminator networks to prevent one from overpowering the other.

Despite these challenges, GANs have shown immense potential across various domains. For example, they are now widely used in creating synthetic data for training machine learning models, enhancing medical images, and generating realistic images in entertainment and advertising. With further research and improvements, such as the development of more stable training methods and better evaluation metrics, GANs are expected to continue evolving and becoming even more powerful and versatile in solving complex problems in artificial intelligence.

In conclusion, GANs have pushed the boundaries of what is possible in generative models. Their applications have expanded far beyond the research lab and into commercial and practical uses. As the field continues to grow, GANs are likely to play an even more significant role in shaping the future of artificial intelligence, making it more creative, efficient, and accessible.

## References

1. Goodfellow I., Pouget-Abadie J., Mirza M., et al. Generative Adversarial Networks // Advances in Neural Information Processing Systems. 2014.
2. Ledig C., Theis L., Huszár F., et al. Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network // CVPR. 2017.
3. Karras T., Laine S., Aila T. A Style-Based Generator Architecture for Generative Adversarial Networks // IEEE Transactions on Pattern Analysis and Machine Intelligence. 2019